

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Effective Scheduling of Energy Consumption in Smart Grids

Jaime Paulo Carneiro Azevedo



Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Henrique Lopes Cardoso (Ph.D.)

July 17, 2013

Effective Scheduling of Energy Consumption in Smart Grids

Jaime Paulo Carneiro Azevedo

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: José Manuel de Magalhães Cruz (Ph.D.)

External Examiner: Álvaro Filipe Peixoto Gomes (Ph.D.)

Supervisor: Henrique Daniel de Avelar Lopes Cardoso (Ph.D.)

July 17, 2013

Abstract

This research focuses on demand side management in *Smart Grids* and the hypothesis of reducing peak demand using *Smart Grid* capabilities.

Alongside with the production of electricity, concerns related with the efficiency of production, distribution and consumption of produced energy appeared. These concerns arise from the willingness of producers to maximize profit and environmental awareness, which is growing everyday in our society.

Driven by that motivation, research in renewable energy resources is increasingly augmenting and potentiating the appearance of new challenges in the production of these cleaner energies, that in addition to be greener are also cheaper in a long term. One of the main challenges is powering all demand with these energies. Renewable energy generators have a long setup time and it proves to be difficult since in peak situations, electricity delivery must be instantaneous, making them dependent on faster delivery time petrol generators to manage peak demands. Managing demand peaks require control of consumer devices which can only be possible nowadays using *Smart Grid* capabilities in order to communicate with consuming devices. This approach also demands a certain flexibility of users to postpone or anticipate appliance executions, having as counterpart cheaper energy prices in certain times of the day.

In this research is assumed that electricity prices are known 24 hours in advance, making it possible to schedule home appliances operation. Therefore, using communication abilities of a *Smart Grid* and electricity prices, this research sets as a main goal to develop an algorithm that can schedule devices in order to help reduce peak demand. This scheduling is constrained by user input, indicating the time frame within which each schedulable device must execute.

The resulting scheduling algorithm is based on a meta-heuristic called *Evolutionary Algorithms*, which uses as a solving technique as a metaphor of human evolution, by trying to mimic crossover between individuals and possible mutations that also happened during human evolution. This method allows to find very good solutions within a reasonable amount of time, making it feasible for a real-world operation. Results are obtained within milliseconds, which for human perception is almost instantaneous.

All goals proposed in this master thesis were successfully completed. Results are promising in terms of employing the proposed algorithm in the production phase of its parent project.

Resumo

Esta investigação foca-se no controlo do lado da demanda em *Smart Grids* e na hipótese de reduzir os picos de demanda de eletricidade utilizando as capacidades de uma *Smart Grid*.

Com o aparecimento da produção de energia elétrica, surgiram preocupações relacionadas com eficiência energética, distribuição e consumo da energia produzida. Estas preocupações advêm dos objetivos dos produtores de maximizar os seus lucros bem como da preocupação com o meio ambiente, que cresce diariamente na nossa sociedade.

Motivado por esses factos, houve um aumento substancial da investigação na área das energias renováveis o que potenciou novos desafios na produção dessas energias, que para além de serem mais amigas do ambiente, são também mais baratas a longo prazo. Um dos desafios é suprir as necessidades energéticas com essas energias. No entanto, o tempo de arranque dos geradores de energias renováveis é longo, o que torna difícil a utilização destas energias em cenários de picos de demanda energética, pois o fornecimento de energia nessas situações deverá ser instantâneo. Assim, nessas situações são utilizados geradores que utilizam energias fósseis são utilizados.

Gerir os picos de demanda energética requer controlo dos dispositivos que estão do lado dos consumidores. Uma forma de tornar isso possível será utilizando as capacidades de comunicação de uma *Smart Grid*. Esta abordagem requer também que os consumidores estejam dispostos a adiantar ou atrasar as execuções dos seus eletrodomésticos em troca de preços de eletricidade mais baixos em certas alturas do dia.

Nesta investigação assume-se que os preços da eletricidade são conhecidos 24 horas antes, tornando assim possível o escalonamento da execução dos eletrodomésticos. Assim sendo, utilizando as capacidades de comunicação existentes de uma *Smart Grid* e o conhecimento dos preços de energia à partida, é definido como objetivo para este trabalho desenvolver um algoritmo que faça o escalonamento dos eletrodomésticos com o objetivo de reduzir os picos de demanda de energia. Este escalonamento de eletrodomésticos é feito tendo em conta as preferências do utilizador, limitando a execução por uma janela temporal que define o espaço onde a tarefa pode ser executada.

O algoritmo criado é baseado numa meta-heurística chamada *Algoritmos Evolucionários*, que utilizam uma metáfora da evolução humana e tentam imitar o cruzamento de indivíduos de uma população bem como possíveis mutações nesses mesmos indivíduos. Este método permite encontrar boas soluções num espaço de tempo razoável, tornando possível em tempo real, encontrar ótimas ou muito boas soluções para cada problema numa questão de milissegundos, situação que para a perceção humana é quase instantâneo.

Todos os objetivos propostos nesta Dissertação de Mestrado foram completados com sucesso. Os resultados obtidos são promissores, pelo que é possível a introdução deste algoritmo, para testes, no projeto mãe.

Acknowledgements

To *my Parents, Jaime e Paula*, for all the support during this years at FEUP, for teaching me how to be a good human being and a man.

To *Avó Ivone*, my first teacher.

To *Ana*, for all the good times, peace, love and friendship that you give me.

To *All my Friends*, wherever you are.

To *Henrique Lopes Cardoso*, for all the patience, support and guidance. You have my deepest admiration and respect.

To *David Moura Ribeiro and Fraunhofer* for having me at Fraunhofer AICOS, for all the support and work environment.

To *Faculdade de Engenharia da Universidade do Porto*, for being my second home during my student years.

Jaime

*“Listen master, can you answer a question?
Is it the fingers, or the brain
that you’re teaching a lesson?
Oh, can’t tell you how proud I am
I’m writing down things that I don’t understand
Well, maybe I’ll put my love on ice
And teach myself, maybe that’ll be nice, yeah”*

Jack White III, Black Math

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Objectives	2
1.3	Main Contributions	3
1.4	Research Context	4
1.5	Document Structure	4
2	Literature Review	5
2.1	Smart Grid	7
2.1.1	Smart Meters	7
2.1.2	Smart Devices	8
2.2	Related Work	8
2.2.1	Active Demand Side Management Using Photovoltaic Panels and Neural Networks	9
2.2.2	On-line and Off-line Scheduling Minimizing Electricity Average Cost . .	9
2.2.3	Scheduling Using a Desired Load Curve as an Objective	9
2.2.4	Real Rime Pricing Using Stackelberg Game Model	10
2.3	Summary	10
3	Effective Scheduling in Smart Grids	13
3.1	Scenario	13
3.1.1	Tasks	14
3.1.2	Schedule	17
3.1.3	Electricity Price	17
3.1.4	Energy	18
3.2	Scheduling Algorithm	20
3.2.1	Evolutionary Algorithms	20
3.2.2	Chromosome and Gene Structures	22
3.2.3	Population Generation	22
3.2.4	Start Time Calculation	23
3.2.5	Crossover	27
3.2.6	Fitness Function	29
3.2.7	Mutation	31
3.3	Architecture	31
3.3.1	Technology	32
3.4	Summary	33

CONTENTS

4	Testing and Results	35
4.1	Input Data	35
4.1.1	Electricity Price	35
4.1.2	Available Electricity	36
4.1.3	Devices	37
4.2	Schedule Quality	40
4.3	Tests	41
4.3.1	Test 1	42
4.3.2	Test 2	44
4.3.3	Test 3	47
4.3.4	Test 4	49
4.3.5	Test 5	51
4.4	Results Analysis	52
5	Conclusions	55
5.1	Future Work	56
	References	57

List of Figures

2.1	Peak, Intermediate and Base Demand.	6
3.1	Smart Grid, Smart Meter and Smart Devices. (Adapted From InterconGreen [aI13])	14
3.2	Task Visual Description.	16
3.3	Examples of a Device's Consumption Curve.	16
3.4	Example of a Schedule.	17
3.5	Example of a 24 Hour Ahead Pricing.	18
3.6	Contracted Power Versus Available Power.	19
3.7	Chromosome Representation.	22
3.8	Gene Representation.	22
3.9	Time Frames Choice.	24
3.10	Roulette Wheel Selection.	27
3.11	Start Time Probabilistic Choice Within Possible Domain.	27
3.12	Architecture of the Main Project.	32
3.13	Architecture of the Scheduler.	33
4.1	Pricing Curve	36
4.2	Electricity Curve	37
4.3	Dish Washer Consumption Curve	38
4.4	Washing Machine Consumption Curve.	39
4.5	Drying Machine Consumption Curve and Task's Preemptive Division.	39
4.6	Algorithm's Output	43
4.7	Available Power and Consumed Electricity by Test 1 Schedule	43
4.8	Results Within 4 Seconds Time Limit.	45
4.9	Optimal Consumption Curve.	46
4.10	Near Optimal Consumption Curve.	47
4.11	Optimal Scheduling Found.	48
4.12	Optimal Consumption Curve.	49
4.13	Results Within 4 Seconds Time Limit.	51

Chapter 1

Introduction

This research focuses on demand side management in *Smart Grids* and the hypothesis of reducing peak demand using smart grid capabilities. The necessity of reducing peak demand in order to increase base demand powered by renewable energy sources is one of the main challenges in the electricity industry for the *twenty-first century*. This research tries to contribute on solving this current problem by aiming at optimizing customer's electricity usage. Scheduling and shifting electricity usage to more advantageous time frames within customer's flexibility and will to help producers. Typically, customers are rewarded with some kind of advantages in order to make up for their flexibility in electricity usage.

After this initial introduction to the problem, in the following section we explain the motivation towards for this research, enumerate the proposed objectives, and point out some obstacles for this work. In the end of the chapter we explain the structure of this document.

1.1 Motivation

Alongside with the production of electricity, concerns related with the efficiency of production, distribution and consumption of energy appeared. These concerns arise from the willingness of producers to maximize profit and environmental awareness, which is growing everyday in our society. These concerns do not only come from producers, but also from governments that are increasingly regulating and legislating energetic sectors. Furthermore there is a growing investment and financing in research sustainable energy [All13a, Sys13, Wee13], specially in the European Union, providing for Intelligent Energy research 730 Million € available from 2007-2013 [Eur13].

When this subject firstly appeared, the initial focus was on production using fossil energy resources. Production took place in central stations that fueled an unidirectional distribution system, which is used even today. In order to minimize losses in this type of systems, research on possible optimizations in the process of electricity generation have been conducted. These researches made possible the increase of produced energy for the same amount of energetic resource [CKW⁺12].

Although these optimizations have improved existing systems, greenhouse gas emissions are still increasing due to the growing consumers' demand of energy from year to year. For this

reason, research for greener energy resources has emerged, from natural gas, hydro, wave and tidal, wind or even solar energy. Many were the energy resources studied for producing energy in order to reduce greenhouse gas emissions. This research has shown that solar and wind energy were the easiest, greenest and production from these sources result in a very low environmental impact [CKW⁺12, CKV11, LSS12].

Despite being cleaner energies, their generation depends on a very specific technology and variable natural factors that make these resources difficult to integrate with the network [ZW11]. This way, these energy resources are less usable unless they can take a significant part of base production. In order to be able to take advantage of this, base demand must be constant, meaning that peak demand must be reduced.

With the appearance of this problem, energy producers began introducing plans for energy consumption where prices depend on energy demand, in order to reduce peak consumption [HG10].

It has quickly become apparent that, even with these measures, there is a need to transform the existing power grid into a “smarter” grid that could offer flexibility, allowing better possibilities of solving the problem. Thus, the appearance of *Smart Grids* started a new path in finding solutions for this problem by enabling exchange of information between the producer and the consumer in order to control and advise consumers about their energy usage[Kri10].

Consumption control brings advantages by serving as a top aide in deviation of consumption to non-peak hours, thus enabling a constant production, bypassing the difficulties in production of wind and solar power [CKW⁺12, CKV11, EMCA⁺11]. This research will be focused on potentiating this consumption control advantages with the utilization of a device, which is present on the *Smart Grid*: the *Smart Meter*. With this technology, consumer appliances’ power demand can be controlled and monitored individually making the energy consumption schedulable. Thus, all consumers will become able to help solving the peak demand, which is the main focus of this work.

Knowing the factors involved in this type of problem, objectives in order to control peak demand must be defined to serve as guidelines during this research. In the next section we explain these guidelines.

1.2 Objectives

To pursue the solution of the aforementioned identified problem, this section points out the goals proposed for this research. Knowing the motivation of studying this type of problems, shown in section 1.1, the main goal is to reduce demand peak in a *Smart Grid* by controlling consumer’s energy use. The hypothesis in this research is:

- Is it possible to create a system that schedules and controls power consumption of household devices, in order to align energy demand with energy production, reducing peak demand?

In order to help finding answers for the hypothesis proposed some specific goals are pointed out:

- To study the best approaches existent in energy scheduling problems identifying the best practices in energy usage;
- To study devices belonging to consumer's side in *Smart Grid's* architecture such as *Smart Meters* and their capabilities;
- To study existent palette of algorithms that schedule power usage in the demand side in order to understand pros and cons of each one;
- To identify the most relevant factors that can influence the scheduling of consumer's devices;
- To develop a software prototype that controls demand on consumer's side and will be running on a low processing power machine called *Smart Meter*.

This work will be part of an National research prototype called EnAware that is being built currently and has another features such as low level communication between appliances and a *Smart Meter*, enabling the software prototype decisions to be put into practice.

Knowing the main goals of this research, the structure of this document and its content is explained in next section.

1.3 Main Contributions

These are the main contributions from this work.

- ***Realism of the addressed scenario.***

This contribution allows scheduling to be more realistic and reduce error when estimating the consumption curve of each computed schedule by giving full emphasis on devices' consumption precision. Also introduce preemptive appliances. With this new perspective on appliances' execution it will be possible postpone some of its parts in order to manage energy consumption or to take advantage of floating prices. Finally introducing a time precision of 1 minute bringing more time accuracy when scheduling. With this, fine tuning each schedule is possible, enabling even more precision in energy consumption.

With this, less energy waste is generated, providing a greener scenario, which is an objective on this type of problems.

- ***Algorithm performance that permits real-time usage.***

Researched algorithm's performance delivers instantaneous and optimal or near optimal solutions, which is a very good progress in this area.

- ***Testing in real life.***

The proposed solution will also be tested in a live test case that will, in the future, testify the advantages of this new approach on reducing peak demand.

1.4 Research Context

This research is part of a national research project hosted by Fraunhofer AICOS named EnAware. EnAware is a project that proposes to electricity consumers a new way of looking to their energy consumption in order to save money and protect the environment. This project contains a vast selection of tools that enable, by using a *Smart Grid*, consumers to control remotely energetic consumption by introducing some usage preferences that enable money savings. This is the module where this dissertation is deeply connected: scheduling appliances in order to reduce peak demand for producers and help customers to save money.

1.5 Document Structure

Besides this first introductory chapter, this document has five more chapters. In Chapter 2 a Literature Review to some relevant and different approaches on this problem is made. These approaches are discussed and pros and cons of each approach are presented.

In Chapter 3 the working scenario is explained in order to contextualize the reader about the different types of tasks. Details about the meta-heuristic used to help solve the problem are also provided, alongside with the motives that ground the decision. Finally, the architecture of the scheduler is explained in full detail.

In Chapter 4 extensive tests are made with a vast palette of scenarios, followed by the results and respective analysis of results. Different type of evaluations, both qualitative and quantitative are made as well as discussion of results.

In the last chapter, Chapter 5, conclusions of this research are presented as well as considerations about future work.

Chapter 2

Literature Review

This chapter studies previous approaches to peak demand control problems, by identifying their context and challenges. Firstly the *Smart Grid* concept is introduced, together with its components, as well as its capabilities and potentialities. It is also shown why this innovative grid is the perfect framework for this problem.

Since the beginning of electricity production, one of the main challenges relates to predict how many energy is necessary at each moment in order to fulfill the needs of consumers at any time. Many researches were performed in order to predict with the most certainty how much energy will be needed from customers in order to avoid energy waste. These predictions are difficult to perform since a considerable amount of energy consumed is coming from random actions performed by users. However some electricity usage is related to the human activities, such as during mealtime or night entertainment, which intervals can be determined previously in order to fulfill that demand. In order to categorize electricity consumption, producers created a nomenclature for the amount of energy needed by customers. Figure 2.1 shows a graph that divides the consumption during day into 3 categories: base demand, intermediate demand and peak demand.

Base demand relates to the minimum energy that must be produced in order to fulfill minimum demand from customers, typically during nighttime. Intermediate demand is normally reached during daytime when the industry's is consuming energy. Peak demand are events that occur when energy demand is substantially higher than expected, making this type of demand a very problematic event to solve. Peak demand normally forces producers to turn on auxiliary energy generators that typically use fossil energies in order to run. These events are very costly to producers because of the rising prices of fossil energy fuels, which also are very harmful for the environment. What if producers could control these peak demand intervals? What if was possible to exchange this fossil energies to environmentally friendly energy sources? These possibilities would also bring economical advantages for producers, since renewable energies' production costs are cheaper.

In order to include renewable energy sources, peak demand must be reduced. Renewable energies' generators do not have the same response time as fossil fueled generators, since they depend on natural factors such as wind or sunlight, therefore they cannot be a solution in emergency sit-

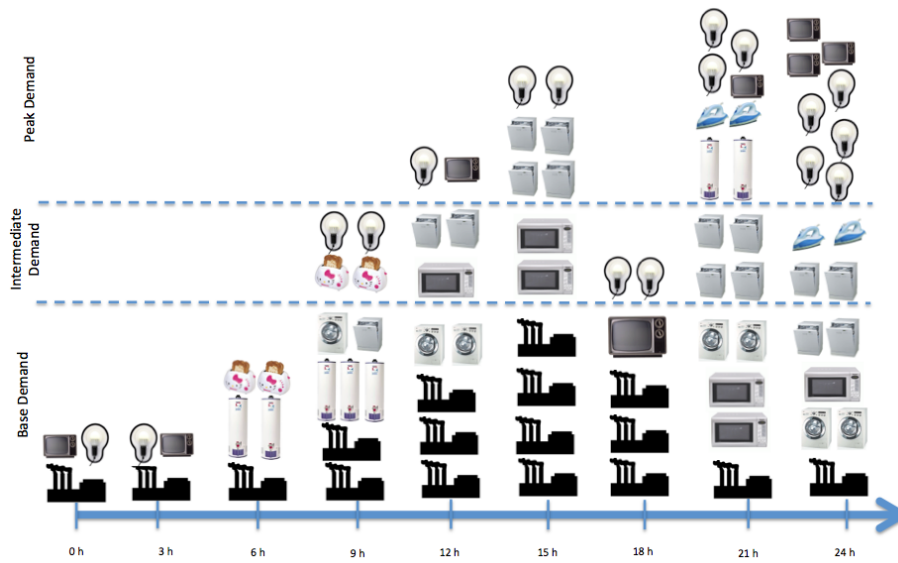


Figure 2.1: Peak, Intermediate and Base Demand.

uations. The solution involves shifting peak consumption to non peak hours, enabling the base demand to be higher and therefore use this renewable energy sources. In order to address this problem, a subject was created in order to study ways of controlling demand named *Demand Side Management*.

Demand Side Management is heavily desired since it brings control in the consumer's side of the grid [Kri10, HG10]. One of the first methods used in electricity industry to reduce demand peaks was *Demand Management by Contract*[HG10]. This contract sets different fixed contracted prices for the energy. Typically this contract imposes 2 or 3 different time intervals, which have different pricing according to producer's expectations [dP13]. The price is higher when the chances of having a peak in demand are higher. With this, electricity companies expected customers to use non-priority appliances in cheapest shifts. Despite being very rudimentary, this method is used nowadays in Portugal [dP13].

This type of demand management has serious drawbacks. Household appliances used nowadays are mainly manual, making them dependent of a human being to operate them. So, this method is very dependent on practicability and willingness to control devices and, consequently, save money on electricity. Thus the non practicability of this method, makes the system unpredictable and dependent [HG10].

The appearance of *Smart Grids* brought a new hope on this subject, bringing automation and planning to consumer's electricity usage. With this, peak demand during all day can be controlled and reduced by applying specific methods such as scheduling, which will be explained later in the document.

2.1 Smart Grid

Smart Grid is a two-way electricity transmission infrastructure that possesses the ability not only of transmitting energy but also information between consumers and producers. The idea behind this type of system is to transform the current one-way power grid into a more collaborative system that enrolls not only producers but also consumers, making energy production and distribution more efficient, reliable and sustainable [Com08].

This improved grid brings a new paradigm for reliability. This paradigm includes a fault detection and self-healing systems that ensure minimum hassle in returning to normal energy supply. With these features, the network can predict power shortage and reroute the power needed. It also can solve energy distribution problems automatically with many techniques such as distributed multi-agent systems [STYP10, Com08].

Bidirectional energy flow is another characteristic of the *Smart Grid* system. It allows energy supply to be not only from the main producer but also from consumers. Consumers have the possibility of connecting their own power supply systems, such as photo-voltaic panels or energy store cells [EMCA⁺11]. This enables the energy company to buy from customers and avoid turning on emergency generators that typically are fossil-energy fueled.

This flexibility featured on this new grid enables the consumer to be an energy seller and so reward the effort and investment made in energetic generation and storage systems. On the other hand, the ability to transmit energy prices helps consumers decide if they are willing to pay the price of electricity on that time or postpone the usage of electricity to when it is cheaper [Com08].

Efficiency and sustainability is brought by the grid's ability to predict and regulate energy demand. With this knowledge production of energy using "greener" sources, such as solar or wind, can be more accurate. Thus production and distribution become more efficient using only the needed energy, evading the loss of unneeded production and increasing in sustainability due to power sources' nature [CKW⁺12, Com08].

Efficiency brought by this new grid requires a consumer's side equipment which manages appliances and their working times. This equipment, called *Smart Meter* makes all the communication between consumers and grid.

2.1.1 Smart Meters

The *Smart Meter* is a powerful and indispensable tool for a *Smart Grid* [DD08]. Along time electric meters evolved from a simple electricity reader to a capable machine that has some processing power. These devices are now designed to be capable of using network connections with other devices in order to communicate with suppliers or smart devices that are using energy. Also *Automatic Meter Reading* (or AMR) enables a whole new range of possibilities in managing the electricity usage in the consumer's side since it can read in real-time energy consumption [Kri10].

With these capabilities the *Smart Meter* is able to control the energy load of all smart equipment connected to it. Each smart device connected to the *Smart Meter* has a programmed threshold. This threshold represents the maximum energy that can be required by each smart device.

When this threshold is reached the *Smart Meter* disconnects the load using its communication capabilities.

Smart Meters also allow the user to control remotely the devices, turning them off and on if necessary. Besides that it allows the producer to communicate the prices, the needs of reducing power usage and other information that need to be known by the *Smart Meter*. These capabilities enable the producer to block consumers' access to electricity due to non-payment almost instantaneously [BZ13].

Distributed electricity generation became possible with this equipment, it may be used to manage when to use the produced energy and when to sell it [Kri10]. This management is possible due to *Smart Meter's* processing capabilities.

Compiling all these features, the combination of communication and processing capabilities brings us to the key point in peak control, making *Smart Meters* the key to manage consumer's side need for electricity. This device also brings the possibility to include specific algorithms that schedule power usage taking into account the user's preferences, postponing usage of non-priority equipments to non-peak hours. This device helps producers to make a more efficient and reliable distribution policy. Another direct consequence is the savings in electricity by the user because typically the off-peak electricity price is lower [Kri10].

2.1.2 Smart Devices

Demand Side Management in Smart Grids implies that all schedulable equipment connected to the *Smart Grid* is able to communicate and receive orders from a *Smart Meter*, which runs the scheduling algorithms. With *Smart Devices*, the system can run without human intervention.

Another key feature of *Smart Devices* is related to the priority of each device on using electricity. This means that some devices can be scheduled, such as washing machines, water heaters or dishwasher machines if the energy is not available or it is not convenient to use it at that time.

Next subsections explain the most relevant approaches studied using these devices in order to schedule their power usage helping reduce peak demand.

2.2 Related Work

After this initial introduction with examples prior to the appearance of *Smart Grids*, literature review reveals some relevant research in *Smart Grid* demand side management. In this section we address to some relevant approaches similar to our hypothesis in order to understand what were the conclusions and the progress achieved in this area.

2.2.1 Active Demand Side Management Using Photovoltaic Panels and Neural Networks

Matallanas et al. [EMCA⁺11] propose an implementation of a active demand side management based on Artificial Neural Networks. The studied system features photovoltaic panels as generators and the scheduling is made to optimize the usage of the generated energy when its production is at peak. So, these neural networks are implemented with the objective of maximizing self-consumption of each individual device connected to the *Smart Meter*. In other words, this scheduler tries to maximize the usage of panel's production in real time since there do not exist storage devices to keep the generated energy.

Concerning to user preferences, this system allows the user to choose time frames when devices can operate; for instance the dish washer may operate from 1 pm to 8 pm. Knowing this, the system predicts photovoltaic generation for the next day and with all this makes the scheduling plan for the equipment to run.

Although this system utilizes a smart meter, it does not communicate with the energy supplier or other producers.

2.2.2 On-line and Off-line Scheduling Minimizing Electricity Average Cost

In [KT11] the problem is addressed in two different off-line modules. The first module is a scheduler that calculates preemptive tasks with a load balancing algorithm that implements a *Valley Filling* method. Non-preemptive tasks are served as a Bin-Packing problem, in other words, the tasks are served as long as there is enough energy to feed the equipment. In this approach the authors use user time and task preferences, task duration and power requirement as objective functions for the scheduler.

Also in [KT11] two on-line demand schedulers are studied. In the first dynamic approach the algorithm simply chooses if a preemptive-type task is served as it arrives or is served at deadline minus the time of the task. The second method named by the authors as *Control Release* has a threshold limit. If the electric usage is below a given threshold the task is performed, if not the task is queued. If electricity usage is not below the threshold until the deadline of the task, the task is performed anyway.

There is a delaying factor that worsens as long as the task is postponed, making the task more likely to be chosen next due to scheduler's preferences.

These four methods (two on-line and two off-line) consider minimizing the longterm average cost. The values used for taking decisions are only related to electricity usage.

2.2.3 Scheduling Using a Desired Load Curve as an Objective

In [LSS12] the authors propose a day-ahead scheduler that uses load shifting as a primary load management method. The objective function is not fixed and system requires a load curve as the model for electricity usage. Also, with the injection of these curves to the system, the

primary objectives of the energy consumption can diverge, from price reduction to peak reduction, depending on user's preferences.

The used algorithm is a Genetic Algorithm that takes the load curve as an objective curve and tries to achieve the similarity between the real curve and the input curve.

During the scheduled day on-line actions are scheduled based on the actual state of the system.

2.2.4 Real Time Pricing Using Stackelberg Game Model

In [CKV11] the authors propose a smart real time pricing (RTP) scheduling using a Stackelberg game model. This model implies that consumers and producers play a role between two possible: follower and leader. In this model, followers answer only after and regarding leader's action.

In this approach the producer leads by setting electricity's price and consumers, *Smart Meters* use electricity pricing to take action. This approach takes into consideration a special type of preemptive smart devices whose consumption can vary during time to help the overall power consumption. As they refer, there are few examples on this type of devices, for example, electric car batteries.

The method uses the *Smart Meter* as a controller / scheduler and tries to minimize each individual appliance usage cost. The scheduler chooses within the time-frame chosen by user, the best time to run each device. Delaying the task from its start implies an inconvenience cost that is taken into consideration in scheduling.

Pricing is defined by the provider as a vector that contains prices for a specific amount of time. Regarding this vector the scheduler calculates the best starting time for each *Smart Device* connected.

Authors refer that this approach stands above day-ahead pricing because it avoids the similarity demand between users which can provoke peaks by the rising electricity demand. Also it relies on real pricing and not in predictions like in day-ahead pricing.

2.3 Summary

All these approaches have shown various ways of controlling users' electricity usage. However there are issues related to this methods. One of these issues is related to communications between producers and consumers: some of the approaches fail to communicate with some time ahead in order to prepare the system to make decisions ahead of time, enabling the system to have more probability of succeeding on scheduling tasks.

Approach [CKV11] has that issue and approach [EMCA⁺11] does not communicate at all. Another issue is related to the type and scheme of input. Approach [LSS12] is an example of a less intuitive input. It uses a load curve to inform the scheduler how the user wants the load curve to be like.

Finally in approach [KT11] and [CKV11] start their method by trying to make approximations of how the grid will behave, leaving a very small room for finding an optimal solution.

Literature Review

For all these reasons this research has proposed a very different approach to this type of problems. The proposed approach is explained in full detail in the next chapter.

Literature Review

Chapter 3

Effective Scheduling in Smart Grids

This chapter will address the work performed during this research. Firstly we explain the scenario and some concepts that are deeply connected to this research. The different task types are explained in order to understand how each one affects scheduling procedures and the final results. Alongside with each type we show adjacent information about each task type.

After these introductory concepts, we explain how a schedule is composed as well as electricity pricing and available electricity concepts, emphasizing how they are deeply connected to the finding of a good schedule.

Next the developed approach, based on the genetic algorithms meta-heuristic, is explained, including the rationale behind this choice. We also show the architecture of the developed system.

3.1 Scenario

This research aims to help control household devices, but since each house has its unique configuration, some notions about this project must be defined in order to understand the prerequisites necessary to enable scheduling and how is scheduling possible with all these possible combinations of household appliances. In this section we present the scenario for this problem as well as some notions which are important to fully understand the subsequent sections.

As Figure 3.1 shows, each household must have access to the *Smart Grid*. Also each house must have an appliances controller and energy meter called *Smart Meter* and some *Smart Devices* which are appliances that can be controlled over the *Smart Grid*, see 2.1.

Each device is connected to the *Smart Meter* that controls and manages electricity requests. With this configuration it is possible to control each device remotely and automatically making not only electricity transmission but also gathering live information about the devices.

Typically, when users need a household appliance to execute a determinate job, they turn on each device at desired start time so they can finish near desired hour in order to help users to fulfill their needs. Undoubtedly, nowadays, with people living an increasingly busy life, the simple action of turning on a washing machine or a dish washer can become very inconvenient. Aside this fact, with the appearance of *Smart Grids* and floating prices, scheduling household devices

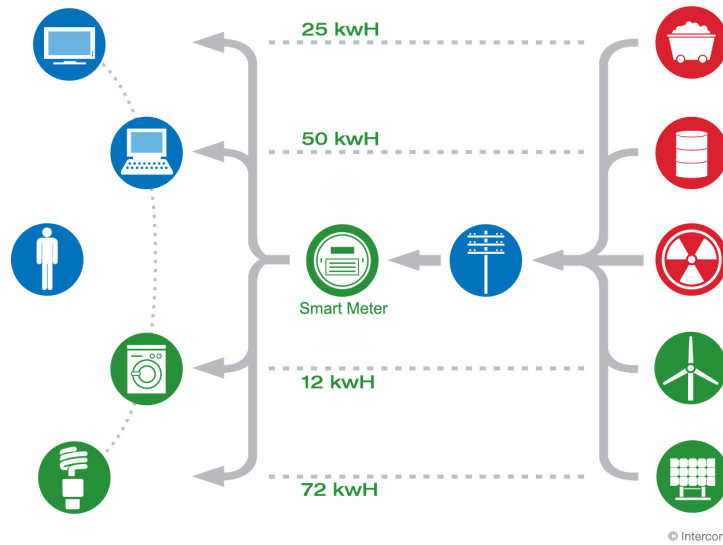


Figure 3.1: Smart Grid, Smart Meter and Smart Devices. (Adapted From InterconGreen [a113])

became a necessity. With this new paradigm consumers can shift each device start time in order to take advantage of cheaper energy. So, what if users can choose, ahead on time, the end time of each household device execution? With this possibility, users can pre-schedule appliances when it is more convenient. In practice, it will be possible to perform instructions to the system such as:

“I want my washing machine to run between 7pm and 8am, the oven to be cooking for 45 minutes and be ready at 9pm and the dish washer to end before 8:45pm.”

With this newer approach to household duties, some challenges appear, such as knowing each device’s execution time for each program offered or even how to define an execution timewise.

Scheduling household devices efficiently in order to reduce electricity peak demand is a key point to explore in this work. For the purpose of this work, these devices can be referenced as tasks, since each one of their operations represents a task with a deadline and execution time for each individual program they run. It is important to mention that not every household device can be scheduled. Knowing this, some considerations and definitions about the different types of tasks must be explained.

3.1.1 Tasks

Naturally there are tasks that must be running as soon as they are triggered, otherwise the user’s experience may be harmed. These tasks are categorized as non-schedulable since they cannot be postponed or delayed fully or partially. Some examples are: lights, using a hairdryer or watching TV.

Despite the non-schedulability of the tasks, they can provide valuable information to the system such as their consumption curve, execution time or even periodicity. This information can be used to make a track of how much energy is likely to be available when scheduling other tasks, as we can see in 3.1.4.

On the other hand, schedulable tasks are more flexible. These tasks can be delayed within user's preferences, enabling the system to choose the best start time for each task regarding the actual price of electricity as well as the usage of electricity at any given time. This scheduling ability help consumers by reducing electricity bill and keep track if the contracted power is truly needed, as well as helping producers and saving the environment by reducing peak demand.

3.1.1.1 Preemptive vs Non-Preemptive Tasks

These types of tasks are schedulable tasks. But they also can be divided into two different groups:

- **Non-preemptive tasks** - This task type allows scheduling but the task cannot be divided. This means that when a task starts it cannot be stopped until full completion.
- **Preemptive tasks** - In addition to be schedulable, this task can also can be divided into several (sub) tasks for more flexibility in scheduling. The division must be defined previously according to machine's internal events. Despite the fact that these tasks can be divided, each sub-task must run after the previous one end.

This research contemplates the two types of schedulable tasks described above. As these two types of tasks can be scheduled, some informations must be known about each task. The information regarding one schedulable non-preemptive task is the following:

- **Baseline** - Earliest possible start of the task;
- **Deadline** - Latest possible end of the task;
- **Duration** - Duration of the task in minutes;
- **Start Time** - Calculated task start time for the best overall schedule found;
- **Consumption(t)** - Consumption of the task in watts, at execution time t , see 3.1.1.2.

Figure 3.2 shows a visual representation of the task and how consumption curve is seen during execution. To represent a preemptive task, there is a need of defining the set of tasks that are parts of the main task. To maintain their sequencing, each task has a reference to its predecessor. This information must be known for each tasks and it is used to prevent malfunctioning of devices. In its most atomic form, preemptive tasks are indeed non-preemptive tasks since their division represents their divided internal working cycles.

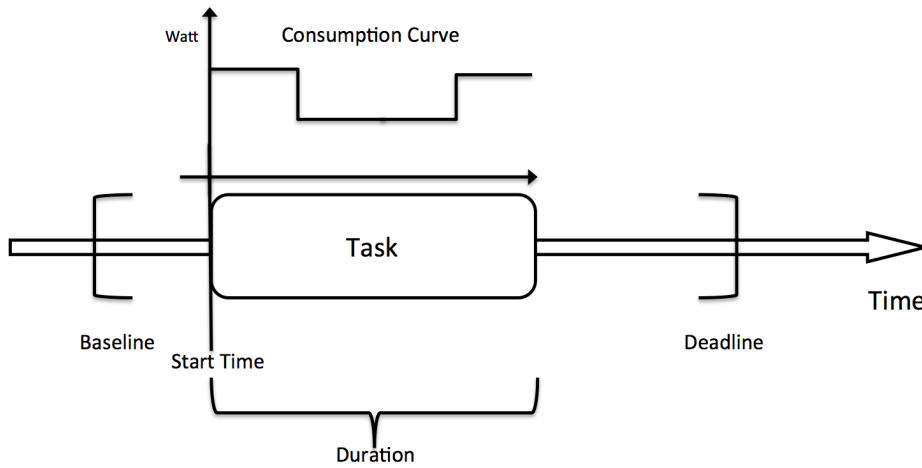


Figure 3.2: Task Visual Description.

3.1.1.2 Task Consumption

In addition to task's timing information, there is one special item that composes a task called *consumption curve*. Each task has a consumption curve associated. This curve represents the electricity usage during the task's execution time. For each program that can be executed on each device, the scheduler must have the information about its consumption in order to enable an accurate schedule's total energy consumption maximizing the authenticity of the schedule. Figure 3.3 shows two possible examples of a consumption curve for a task.

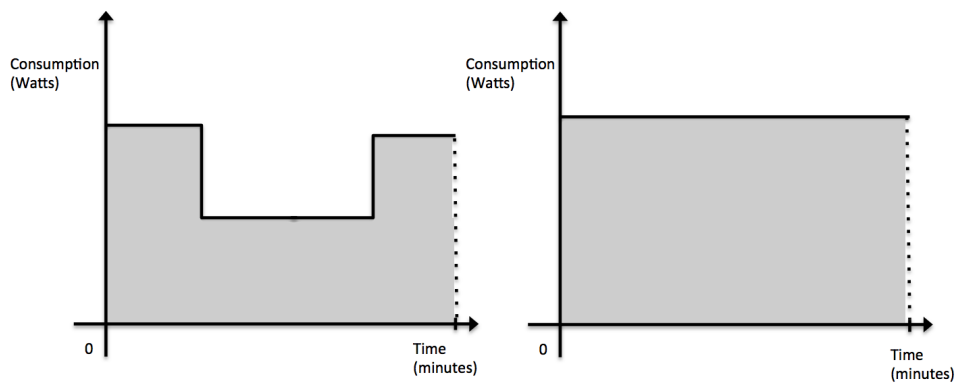


Figure 3.3: Examples of a Device's Consumption Curve.

There are some devices that possibly can have this variable consumption curves such as dish washers or washing machines that during some intervals of their executions use more energy, for example, when heating water or less energy, when flushing the water out of the machine. Other devices such as microwaves have a constant consumption curve, meaning that the consumption in every moment of the execution is the same. After this definition of all schedulable tasks, the next

subsection shows how these tasks can be combined and defines some characteristics of a set of tasks.

3.1.2 Schedule

A schedule is a set of tasks that are combined in order to meet user's preferences for all devices. For the purpose of this work, a schedule is valid if it fulfills the following criteria:

- The schedule contains all schedulable tasks within a time frame of 24 hours;
- All tasks must be executed and terminated within its baseline and deadline;
- No part of a preemptive task can have its start time before end time of its previous part;
- Energy consumption cannot be above the available power at each moment.

Atomic time unit is 1 minute, meaning that schedule precision is within 1 minute time.

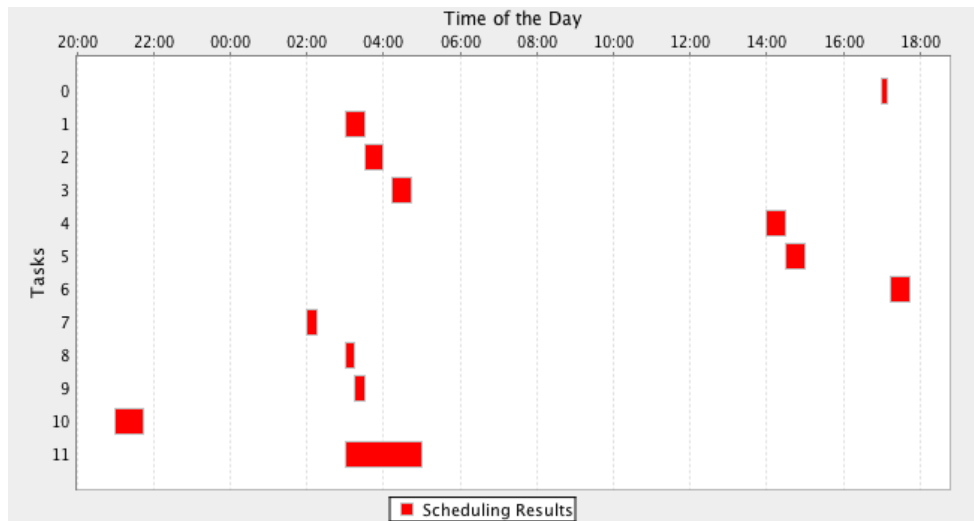


Figure 3.4: Example of a Schedule.

In figure 3.4 we can see a possible schedule for one day. One valid schedule is better than another if its total cost is below the other's cost, meaning that the lowest cost, the better schedule. In order to know the cost of each schedule there must be the information about electricity prices at each point in time. Also, electricity prices are used to calculate the start time for each task. This issue is addressed in the next subsection.

3.1.3 Electricity Price

One of the main features of a *Smart Grid* is the ability to communicate data in addition to power transmission between producers and consumers. This data connection makes possible real time

information between all terminals connected to the grid. One of such informations is price of energy.

Literature review [KT11, CKV11] has shown that transmission of energy prices can be performed in various ways, being the two more relevant the 24-hour-ahead pricing and real time pricing. In this research 24 hour ahead pricing was chosen as a main pricing announcement policy, because it enables the scheduling system to have a decent time window to schedule the tasks and plan the best schedule. This makes scheduling easier and with better results since scheduler knows beforehand what are the prices and when they will be in place, making it possible to maximize profit. On the contrary, real time pricing would lead to unexpected price changes and consequently less flexibility to better adapt the schedule to changed prices since not every running task can be stopped.

Another key information pricing gives is that, pricing for the next 24 hours resembles producers' intention on where to reduce electricity demand [RW08]. To achieve that, producers increase the price of kW/hour on some time frames and decreases in other ones in order to reward customers that have flexibility to change their demand from peak to non-peak hours. In this research the price of energy is the key factor to perform scheduling since it is the only factor on the whole *Smart Grid* system that informs customers on producers' intent of reducing or increasing electricity consumption.

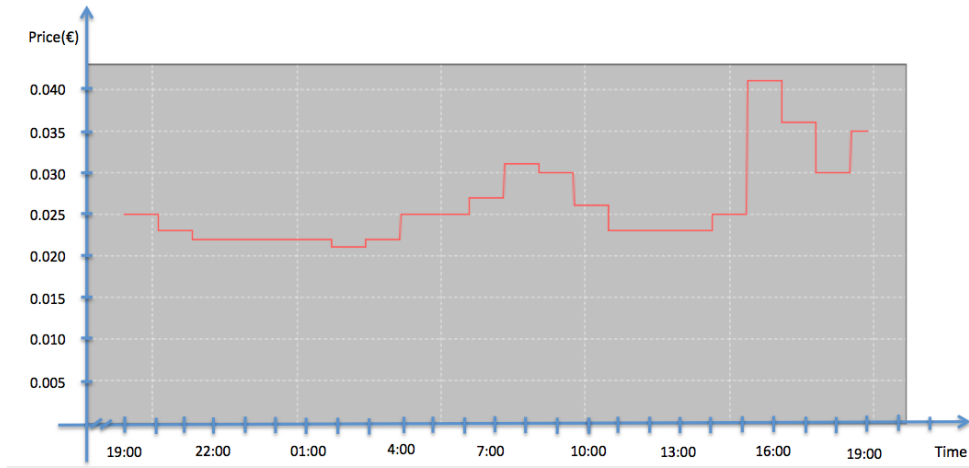


Figure 3.5: Example of a 24 Hour Ahead Pricing.

Figure 3.5 shows one possible pricing curve for the next day's scheduling.

3.1.4 Energy

Another essential piece of information for scheduling concerns the but also availability of energy. Electricity supply is limited. Each household typically has a contracted power limit. Contracted

power is the maximum electric energy available for a specific customer. Typically producers provide several contracted power limits so customers can choose the one that better suits up more to their necessities. When the contracted power limit is exceeded, energy supply is automatically interrupted, shutting down all electric devices. This contracted power must cover both schedulable and non-schedulable tasks. This makes more difficult to predict how much power will be available since non-schedulable tasks are more difficult to predict. Sharing the power with non-schedulable tasks also narrows the range of possible solutions and can make scheduling more difficult.

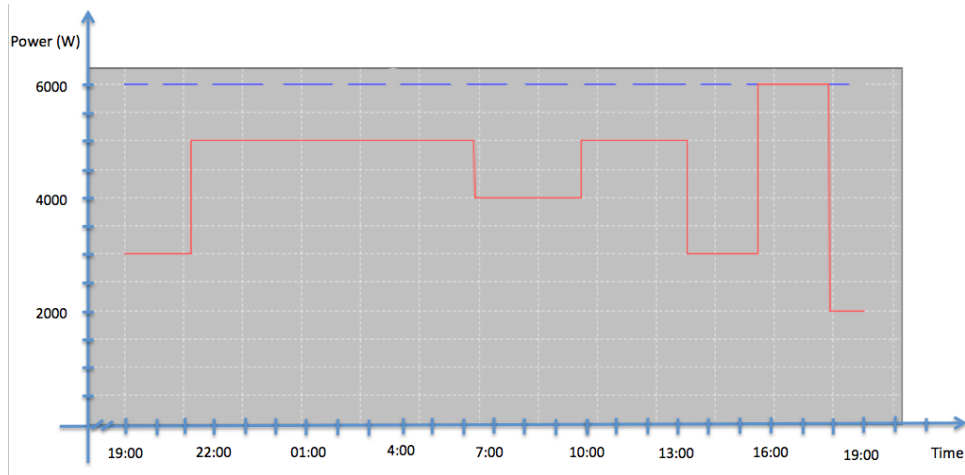


Figure 3.6: Contracted Power Versus Available Power.

Figure 3.6 shows an example the correlation between contracted and available power. The dashed line shows the contracted power between customer and producer and the continuous line represents the available power for running schedulable tasks. There is no way of predicting with 100% confidence the energy that will be used by customers on non-schedulable tasks. This consumption depends on human behavior that can be very random. Other factors that enable more random results include temperature, because it affects the way the climatic devices are used, or sunlight, that affects artificial light usage. There are many possible approaches to deal with this problem such as forecasting algorithms, data mining or machine learning [MM09, TdMM06].

Since prediction non-schedulable energy consumption is not the main purpose of this research, we will assume that available power is given as input for the scheduler. This input can be calculated as an average of previous 72 hours non-schedulable devices' consumption or another method that suits this purpose.

In each minute of the day, available power is contracted power minus the average of used power on non-schedulable tasks.

After revising all factors that compose or can influence a schedule, we now turn to explaining the scheduling algorithm proposed, including the meta-heuristic adopted.

3.2 Scheduling Algorithm

This section contains all the information related to the scheduler created, including detailed information of each one of its components. Literature review has identified, besides different approaches to this type of problems, some algorithms that could be used as a starting point to this problem. In this research was chosen to create a task scheduler using *Evolutionary Algorithms*.

Evolutionary Algorithms are meta-heuristics that try to mimic the biological evolution of species. Inheritance, crossover, mutation and selection are the key techniques that are computationally replicated in order to reach near optimal solutions.

In this research, there are some reasons why evolutionary algorithms were chosen. One of the shortcomings of this type of algorithms is that there is no guarantee for optimality. In this particular case, an optimal solution is not required since the quality of each solution is related to billing of energy. Being a soft constraint, price is not in critical matters such as task precedences or exceeding the available power, which are the hard constraints in this problem. Also, there are many possible solutions in each task that despite not being optimal bring noticeable reductions in the final price of each schedule.

Another reason for choosing evolutionary algorithms is the fact that the search space is very large and complex, due to the power, price, precedence constraints and time granularity being 1 minute. For instance, considering a schedule for 3 simple, non-preemptive tasks their number of possible start times is presented in Table 3.1.

	Baseline	Deadline	Duration	Number of possible start times
Task 1	0 min	60 min	10 min	$(60 - 0) - 10 = 50$
Task 2	0 min	120 min	60 min	$(120 - 0) - 60 = 60$
Task 3	0 min	45 min	5 min	$(45 - 0) - 5 = 40$

Table 3.1: Number of Possible Start Times for a Simple 3 Task Schedule.

If the three tasks shown in Table 3.1 are to be combined into the same schedule, they could generate 60000 different start times for the search space, turning a small sized problem into a relatively big domain.

Being a stochastic and parallel meta-heuristic, an evolutionary algorithm helps the system not to stuck on local-optima solutions and enables the possibility of finding good solutions very quickly. Another reason for selecting evolutionary algorithms is related to the vast array of researchers on scheduling problems that have successfully used this approach.

3.2.1 Evolutionary Algorithms

As previously mentioned Evolutionary Algorithms emulate biological behavior in order to reach the best solution possible for each problem to be solved. Typically, an evolutionary algorithm is composed of a *population of chromosomes* characterized by genes. A *fitness* function is used to evaluate each *chromosome*, which are combined through a *crossover* function and altered using a *mutation* method. Figure 1 shows the algorithm that explains one execution of an Evolutionary

Algorithm.

Algorithm 1 Evolutionary Algorithm

```

1: GeneratePopulation(NIndividuals)
2: while (BestIndividual.Fitness < MinimumFitnessRequired) do
3:   for all (Individual in Population) do
4:     CalculateIndividualFitness
5:   end for
6:   ChosenIndividuals  $\leftarrow$  Choose2Individuals()
7:   NewIndividuals  $\leftarrow$  Crossover(ChosenIndividuals)
8:   if (ProbabilityOfMutation()) then
9:     Mutate(NewIndividuals)
10:  end if
11:  ReintroduceIndividualsIn(NewIndividuals, Population)
12:  BestIndividual  $\leftarrow$  MaxFitnessIndividual(Population)
13: end while
14: return BestIndividual

```

Initially the algorithm starts by generating a population of N individuals as chromosomes, that will be the first set of elements analyzed and evolved. This generation can be random or follow some kind of heuristic to achieve good initial starting points, which compromise scheduling solutions. The initial population is very important since it enables reaching good solutions: the best the initial population is, the more probability exists in the appearance of better and faster results. This generation is only used one time for each execution of the Evolutionary Algorithm. After generation and in each iteration of the algorithm, each individual is checked for its fitness value that measures how good is the individual.

This fitness value is the differentiating factor between individuals. Pairs of individuals are chosen to be crossed-over and breed new individuals that hopefully have better fitness value than their parents. The crossover is performed by combining “parts” from each parent to generate two new elements that will be part of the next generation of the population.

After crossover, mutation of generated individuals typically occurs within a certain, often small, probability. This mutation consists in changing one gene of the chromosome to another state in order to boost the appearance of even better solutions.

This type of algorithms run for a certain number of iterations, each generating a new population. At the end, the population is checked and the best chromosome, regarding fitness, is returned.

The next sections explain in depth the application of the evolutionary algorithms with every element in this research.

3.2.2 Chromosome and Gene Structures

The chromosome is the computational representation of an individual - a schedule - which is composed of a scheduled set of tasks. A chromosome is one individual in the population and the most fit chromosome is the best schedule for a given set of tasks.

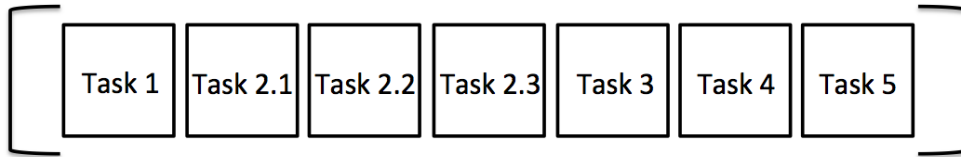


Figure 3.7: Chromosome Representation.

Figure 3.7 shows a possible representation of a chromosome that contains a set of genes. A gene is the smallest representation in an Evolutionary Algorithm. In this research, it represents a schedulable, non-preemptive task because at this time even preemptive tasks are divided into its sub-tasks, which are, as said previously, non-preemptive. Within this gene there is all the information regarding the task. Details about the information contained within each gene can be found in 3.1.1.

One very important value contained into the gene is the tasks' start time. It is crucial that the start time of the task is calculated in a good direction because the overall fitness of the schedule depends on this value. Subsection 3.2.4 explains with detail the generation of a task's start time within a specific schedule.

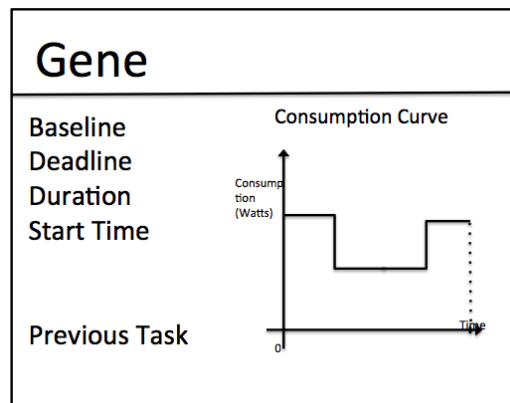


Figure 3.8: Gene Representation.

3.2.3 Population Generation

In this research the generation of the population is straight forward, generating a total of 100 individuals. Initial population has a minimum 70% of valid individuals to enable faster results,

avoiding having a majority of invalid individuals as much as possible. While avoiding this majority, some invalid individuals can also be useful in improving the solution, given the evolutionary nature of the algorithm. This also enables a good balance between the quality of the population and generation time, since providing valid chromosomes can be much slower than randomly providing one (valid or invalid), depending on the number of genes in the chromosome.

The generation of the initial individuals depends on pricing and tasks to perform. With this information, individuals can be generated, gene by gene with the knowledge of each task's information to calculate its start time. The calculation of each new start time is explained in depth in subsection 3.2.4.

3.2.4 Start Time Calculation

In order to reach the optimal solution, each task calculates its own start time using the pricing intervals given to the scheduler. After receiving the pricing array, a rather complex algorithm is used to choose, stochastically, the best start time for the task. It starts by checking if the task has any preceding task, to avoid overlap between chunks in preemptive tasks. Note that the first sub-task in each preemptive task is recognized as a non-preemptive task, since it has no predecessor, so its start can be the task's baseline. At this initial point every task has its time window ending at the deadline.

After obtaining the first time boundaries found, and in order to avoid further unnecessary processing, if the available time for running the task is equal to the task's duration, the start time will be equal to baseline.

Until this point, there is not much calculation since time restrictions do not allow any heuristic. After this stage, start time calculation requires aid from the electricity pricing to calculate which start time should be used. As mentioned previously, electricity pricing plays a role on informing when producers want the customers to use energy in order to avoid peak demand. This could be easily achieved by choosing the cheapest pricing frame within the task's baseline and deadline. However, these actions can lead to a problem of energy starvation since the power required for running all tasks that can be executed within that price time frame, might be higher than the available electricity.

In order to avoid electricity starvation, each task must know what are the pricing frames where it can be running.

Narrowing the domain to only valid time frames, as seen in Figure 3.9 enables performance, but since it is very simple, something more must be done. Before going further into the computation of a task's start time, we must check if the task is only eligible for one price frame.

3.2.4.1 One Pricing Frame

On pricing frame means that due to task's time specification and pricing time frames, the start time can only be chosen into only one electricity price. Therefore, there is no argument in differentiating each possible start minute because, in terms of pricing, they represent the same price. So, at

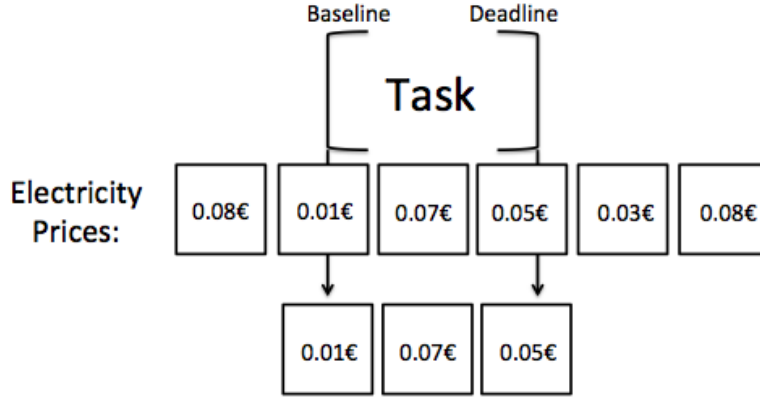


Figure 3.9: Time Frames Choice.

this point the only concern is related to fine tune each start time in order to enable a best overall schedule. For instance, each start time can lead to different electricity consumption curves that can exceed the available power. In order to take all this into consideration, for one pricing frame we run the following algorithm:

Algorithm 2 Start time calculation with one pricing frame.

```

1:  $start \leftarrow thisTask.Baseline$ 
2:  $end \leftarrow thisTask.DeadLine$ 
3: if ( $thisTask.hasPreviousTask$ ) then
4:    $start \leftarrow previousTask.EndTime$ 
5: else
6:   if ( $random(10) < 5$ ) then
7:      $StartTime \leftarrow start$ 
8:      $StartTime \leftarrow start + randomMinute(end - start - thisTask.Duration)$ 
9:   end if
10: end if

```

Algorithm 2 shows that in, 50% of the times the start time is the task's baseline or preceding task's end time, if the task is part of a preemptive task. The other 50% of times a random start time is chosen within the possible minutes without ending the task after deadline.

3.2.4.2 More than One Price Frame

Having more than one pricing frame brought the need for choosing between the possible ones. In this research a probabilistic method is used that enables choosing between the different prices. This method is often called *Roulette Wheel Selection* [LL12, Cen13]. This heuristic enables a probabilistic choice between possible prices in order to favor cheapest pricing time frames. This choice was made because of performance issues. As said previously, since the time unit chosen

was 1 minute, search space can be big and consequently make schedule search slow. Knowing this, there is more chance in finding the best schedule by assigning each task to cheapest pricings. This cannot be hard-ruled since it can lead to a probable state of energy shortage and impossibility of performing schedules. Instead, *Roulette Wheel Selection* makes possible to address with higher priority the cheapest price without compromising power validation, making possible the choice of a second cheapest price and so on, if needed.

In order to choose the starting time the ratio between each price and the sum of all pricing values is calculated. These ratios are used to give a probability of each interval to be chosen. Higher probability values are given to cheapest pricing intervals. The following example illustrates this process.

For one given task, the possible price frames are the following:

0.1€ / kW/h	0.3€ / kW/h	1.1€ / kW/h	0.5€ / kW/h
-------------	-------------	-------------	-------------

After identifying the possible pricing frames, the weight of each one within the sum of all prices is calculated. Formula 3.1 calculates the weight value of each pricing.

$$weight(price_i) = \frac{price(i)}{\sum_{t=1}^{NPrices} price(t)} \quad (3.1)$$

For this example, the result of the previous equation will result in the following.

Price	0.1€ / kW/h	0.3€ / kW/h	1.1€ / kW/h	0.5€ / kW/h
Weight	0.05	0.15	0.55	0.25
Probability	55%	25%	5%	15%

In order for these weight values to be applied as the probability of each price to be chosen, they must be assigned to each price sideways, assigning to the cheapest price, the weight of most expensive price frame. This assigning operation is performed by an auxiliary algorithm. Figure 3.10 shows the metaphor behind this process.

After choosing a price frame, start time calculation checks if the duration of the task is equal to the pricing duration. If true, the returning value is the pricing's start time. If not, the start time calculation proceeds by calculating if the task duration is not higher than pricing's duration. If it is, tasks's start time is calculated by using all of pricing duration and 50% of times it chooses to anticipate start time to previous pricing, while other 50% of times it chooses to delay to next pricing. If the task is still bigger than available time frame, it repeats the same decision until the task fits into the time frame. This proceeding occurs only if task's baseline or deadline are not violated.

On the other hand if the task's duration is smaller than the pricing duration, the decision is divided by 3 methods. The first one returns the later time between task's baseline and pricing start time in 50% of times. With 35% of chance is chosen a random minute within the first quarter

Algorithm 3 Start Time Calculation For More Than One Time Frame.

```

1:  $price \leftarrow RouletteWheel(AllPrices)$ 
2: if (start < price.StartTime) then
3:    $start \leftarrow price.StartTime$ 
4: end if
5: if (end > price.EndTime) then
6:    $end \leftarrow price.EndTime$ 
7: end if
8:  $domain \leftarrow end - start - thisTask.Duration$ 
9: if domain < 0 then
10:
11:   if end - domain <= thisTask.Deadline && start + domain >= thisTask.BaseLine then
12:
13:     if random(10) < 5 then
14:
15:       return end - domain
16:     else
17:       return start + domain
18:     end if
19:   end if
20: else
21:
22:   if (end - domain <= deadline) then
23:
24:     return end - domain
25:   end if
26:
27:   if (start + diff >= liveline) then
28:
29:     return start + domain
30:   end if
31: end if

```

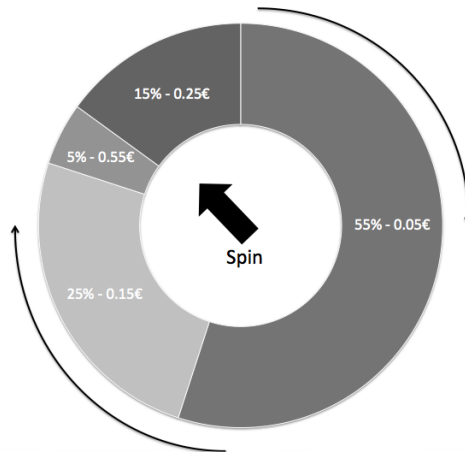


Figure 3.10: Roulette Wheel Selection.

of domain. The final 15% of times, start time is chosen as a random minute within domain. Figure 3.11 shows a visual representation of which probability is assigned to each part of the task.

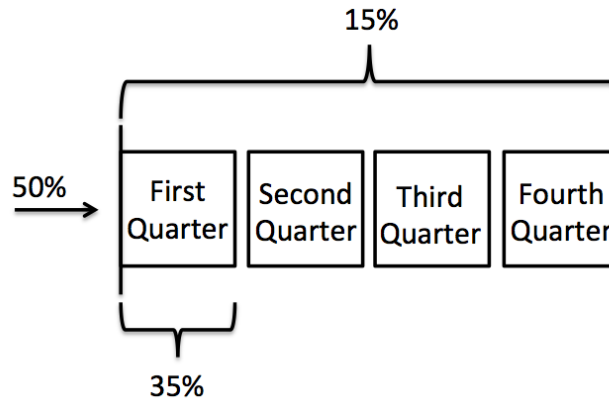


Figure 3.11: Start Time Probabilistic Choice Within Possible Domain.

Concluding, this last nuance of calculating start time is implemented by following algorithm.

The performance of this algorithm is deeply connected to maximizing overall performance of the whole scheduling algorithm since it focus on giving more importance to some time chunks of the domain, which they are more likely to contribute to a best overall schedule.

3.2.5 Crossover

After population generation, the evolutionary algorithm starts manipulating individuals in order to generate fitter individuals. One of the operations performed is called crossover. This crossover operation resembles biological crossover by creating new individuals with genetic combinations of its parents. Computational crossover relies precisely in the same method, combining fittest genes from each chromosome, generating a better individual.

Algorithm 4 Start Time Calculation If Task's Duration is Smaller Than Pricing Duration.

```

price  $\leftarrow$  RouletteWheel(AllPrices)
if (start < price.StartTime) then
    start  $\leftarrow$  price.StartTime
end if
if (end > price.EndTime) then
    end  $\leftarrow$  price.EndTime
end if
domain  $\leftarrow$  end – start – thisTask.Duration
if domain >= TWOMINUTES then

    if (random.nextInt(10) < 5) then

        return start
    else
        if (random.nextInt(10) < 7) then

            return random(firstQuarterOf(domain))
        end if
    end if
else
        return start + random.nextInt(domain)
    end if

```

There are many computational methods to perform crossover such as one-point crossover or two-point crossover [JS92]. Despite the existence of these methods they did not fit as a first choice for this problem. One and two point crossover need many iterations to create new chromosomes that can be better than its parents in this particular problem. Also they can easily violate available power and task precedences.

For the sake of this research, another non elitist crossover method was created in order to enhance performance. This crossover method speeds up the generation of good chromosomes by interlacing all genes to form two new chromosomes. After Crossover the first child chromosome contains the cheapest genes between the two parents and second chromosome receives the most expensive genes. At this stage no power or precedence validation is made. In case of existing preemptive tasks, each generated chromosome is analyzed and for each preemptive task that starts before its predecessor is calculated a new start time to prevent precedence failures. After that, two fittest chromosomes from initial four are chosen and reintroduced into the population.

The crossover method is repeated, combining the most fit chromosomes in order to help finding the optimal solution for each scheduling problem. One of crucial factors for crossover to be successful resides in the choice of parent chromosomes since they are the origin of child chromosomes that bring better solutions.

3.2.5.1 Chromosome Selection

There is higher probability of finding very good solutions if the right chromosomes are chosen to be crossed. The method chosen for selection in this research is not very common but since the results must be reached within a small time frame, more complex choosing methods must be avoided.

Selection in this research is performed in a very simple way. When choosing candidates, there are 50% chances of choosing the best individual to cross with another random individual. In other 50% of times two random individuals are chosen. With this random choice computational expenses are low, since there is no need to calculate any type of probabilistic choice regarding the fitness of each chromosome in the population. Also, improving the best chromosome with other random chromosomes has shown to be a very good vehicle to reach optimal solutions, from the fact that search space can be large.

Next, the fitness evaluation function that quantifies how good is each individual.

3.2.6 Fitness Function

The fitness function is used to analyze chromosomes and measures its level of “goodness”. With this feature it is possible to compare or select chromosomes during the whole execution of the evolutionary algorithm. First of all, there is the need to define what factors are important and contribute in each chromosome for its fitness. The fitness function defined for this research takes into account three factors:

- **Schedule Price** Sum of every task cost, taking into account the prices during task’s execution time;
- **Task Precedences** The number of precedence failures on preemptive tasks;
- **Power Usage Validation** The number of minutes that consumption of all tasks is above the available power.

In order to deeply understand the fitness function, each factor is explained next.

3.2.6.1 Schedule Price

One of three factors in fitness function is the cost of each task. Since price of energy is not constant, neither task’s start time, each task will have a different cost on each different schedule. With this being said, equation 3.2 calculates the cost of one entire schedule.

$$Cost = \sum_{i=1}^{AllTasks} \left(\sum_{t=StartTime}^{StartTime+TaskDuration} TaskConsumption(t - StartTime) * Price(t) \right) \quad (3.2)$$

The cost is calculated by adding each multiplication of task's consumption at each minute and price of energy at each minute of the day that the task is running.

3.2.6.2 Task Precedences

Another factor that contributes for the fitness value is preemptive tasks' parts precedences. This factor is only taken into account if the schedule figures preemptive tasks. All preemptive tasks are *Finish to Start*, meaning that each task start time depends on previous task end time. There also can be a lag > 0 between each task. In order to guarantee this rule to be met, algorithm 5 validates it.

Algorithm 5 Task's Precedence Calculation.

```

for all Tasks in Chromosome do
  if (Task.HasPreviousTask) then
    if (Task.PreviousTask.EndTime > Task.StartTime) then
      PrecedenceErrors++
    end if
  end if
end for
return PrecedenceErrors

```

3.2.6.3 Power Usage Validation

Power usage validation checks if in any scheduled minute the power usage, being the sum of each task consumption in that minute, is exceeded. Alike task precedences, this factor is only taken into account if the sum of maximum energy consumption of each task, between their baseline and deadline, is below the available power.

```

for time = FirstTask.StartTime to LastTask.EndTime do
  consumption  $\leftarrow$  0
  for all Tasks task in Chromosome do
    if ((time >= task.StartTime) && (time < task.EndTime)) then
      consumption += task.getConsumption(time)
    end if
  end for
  if (consumption > AvailablePowerAt(time)) then
    ConsumptionErrors++
  end if
end for
return ConsumptionErrors

```

The fitness function combines these previous explained factors in order to generate a value that categorizes each chromosome. Despite of being electricity the real cost for the customer, it is the least important in the fitness function because the other two factors are heavily penalizing since if their value is above zero in any of those factors, the schedule will be impossible to execute.

Knowing this, their value must be much greater than the electricity cost to prevent the evolutionary algorithm to pick impossible solutions. The fitness value is calculated by the following equation, shown in 3.3.

$$fitness = ScheduleCost + ConsumptionErrors * W_C + PrecedenceErrors * W_P \quad (3.3)$$

W_C is a weight value that aggravates the fitness of faulty chromosomes when they present consumption errors and W_P aggravates the fitness value when task precedences are not complied with. These weight values represent different magnitudes in fitness, giving more importance to an error if it is a consumption error. Therefore $W_C < W_P$. In this research, the value for W_C was 10000 and for W_P was 1000.

3.2.7 Mutation

The mutation function is very relevant in this work. Typically, this function plays an important role in evolutionary algorithms. In this work it is used to change the encoding of one chromosome by modifying one gene in order to try and reach better solutions. This is related to the search space in this scheduling problem. Sometimes solutions require that some tasks use their most expensive time frame in order to compute a valid solution. As seen before, the probability of choosing less cheap pricing intervals is reduced by roulette wheel selection. The mutation function is used to solve this problem as well as speeding up the search of good solutions by re-calculating the start time for each mutated gene.

In our approach, the mutation is used 10 times on each new breed of a population. On this work mutation consists in recalculate each gene start time. Also this mutation function is greedy by only changing gene's start time if the overall fitness becomes better. With mutation we try to address also invalid schedules that can become very good solutions by changing one gene.

3.3 Architecture

In this section we explain the architecture of the scheduler as well as the interaction between the different existing modules. This scheduling module communicates with other existing modules in the main project entirely through an interface that manages all the input and output information between our module - Scheduler - and main project's services as depicted in 3.12.

Figure 3.13 shows how each component is related to others as well as identifying which services are needed by the evolutionary algorithm in order to compute each schedule. There are three external services that provide information needed by the scheduler, these are "Power Analytics", "Device Information" and "Electricity Price".

The "Power Analytics" service provides all information related to electricity quantification. This service allows the scheduler to know how much is the maximum power as well as power

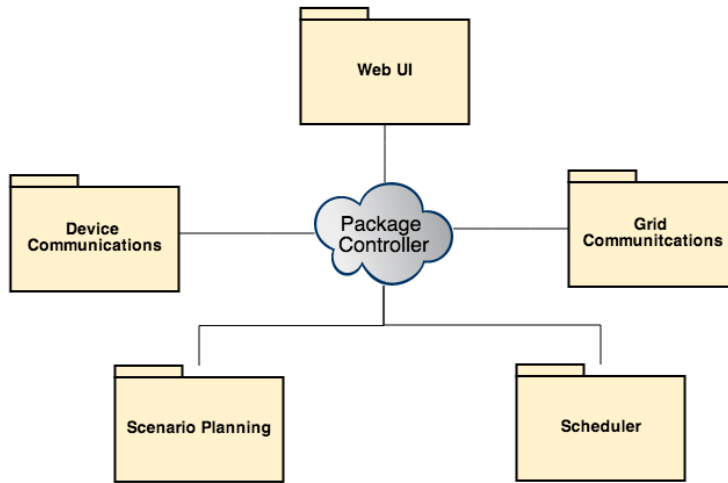


Figure 3.12: Architecture of the Main Project.

available for schedulable tasks. This information is deeply needed for chromosomes when calculating fitness values in order to understand if a schedule uses more energy than available at every minute.

“Device Information” provides all information about each device present in the schedule. This service provides the execution time and consumption curve for each program on each device. It also provides information about preemptive tasks and their timing in order to avoid malfunction of devices or less effectiveness on performing their job. Without this service it would be very difficult to provide a real schedule since the knowledge of execution time for each task is essential in order to not infringe the deadline for each task.

The last service is “Electricity Price”. This service provides the price of kWatt-hour for any-time within the next 24 hours of execution. This is used by “Task” not only to calculate task’s price but also to calculate start time of each task, which is one of the most important functions of the scheduler since it enables finding the best start time values for each task in schedule.

The “Evolutionary Algorithm” module combines all these elements in order to pursuit the best schedule possible for each input. From this module all information is returned to the interface that provides the computed schedule to other EnAware modules so that the manipulation of devices can be performed.

3.3.1 Technology

This module was implemented using Java OSGi. This technology allows the developer to build the software as modules that communicate between themselves using services. With this dynamics OSGi framework allows the modules to be updated on runtime, which is very advantageous since this system needs to be always on-line. The system must be as reliable and available as

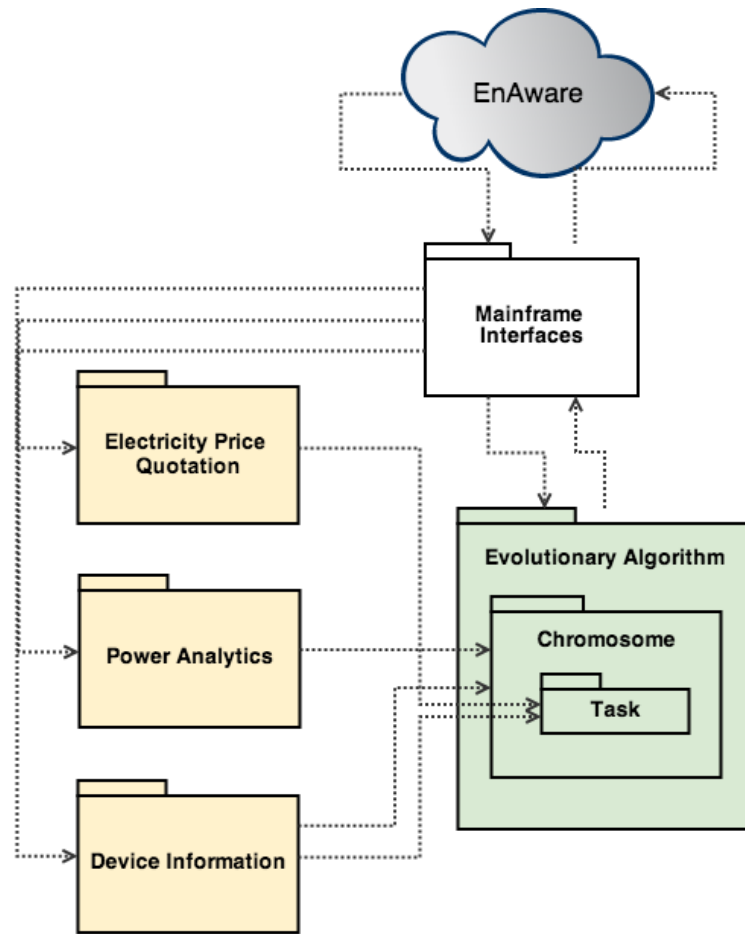


Figure 3.13: Architecture of the Scheduler.

possible because electricity usage is continuous and so must be the associated applications. This modularity allows the developer to reuse modules, adapt modules and manage them in an easy way, making the development process easier and less error-prone [All13b].

3.4 Summary

This developed scheduling algorithm is believed to be the best approach to this problem by combining fast scheduling with money saving schedules and so, potentiating the reduce of peak demand which is the main goal of this research. This is achieved by combining very simple heuristics that can reach very good solutions with not much expense of computational power. The next chapter addresses the veracity of this arguments, in order to acknowledge these method as a appropriate method for energy consuming scheduling.

Chapter 4

Testing and Results

In this chapter we address a group of tests in order to validate the solutions given by the scheduler developed during this research. To guarantee that each test case is validated successfully and robustly each scenario is ran many different times in order to understand how this algorithm performs and speeds when scheduling.

Also, in this chapter we present the input data for all tests. Since there are many possibilities of testing, some input data which is considered relevant is tested so it can lead to many different schedules only with different devices. These input data are the pricing curve, which relates to the price of energy on each different time of the day, the available electricity to be assigned for schedulable tasks and the set of different devices used to generate the tests, each one having its own specifications in terms of task duration, energy consumption and preemptiveness.

4.1 Input Data

Input data informs the system about resources like electricity prices 4.1.1 and electricity availability 4.1.2. Other information is related to which devices must be scheduled 4.1.3. This data determines the quality of obtained schedules.

Not taking into account available power is likely to obtain infeasible schedules, while ignoring energy prices does not allow us to obtain near-optimal solutions in terms of overall energy consumption costs.

Also, pricing information was set as described in 4.1.1 and available power is calculated regarding user's power usage history.

This test-case scenario is made from 7 pm of a given day. The electricity cost and available power related to this scenario is known for a 24 hour period, that is, until 7 pm of the following day.

4.1.1 Electricity Price

The electricity price curve is the information retrieved from the grid, in a 24 ahead basis, about the price of kW/h on each moment for the next 24 hours.

Testing and Results

On this batch of tests, the pricing curve used is from a real smart grid system located in Australia on August/2012 which is a summer day. Figure 4.1 shows a 24 hour pricing curve, between 7 pm and 7 pm of the next day.

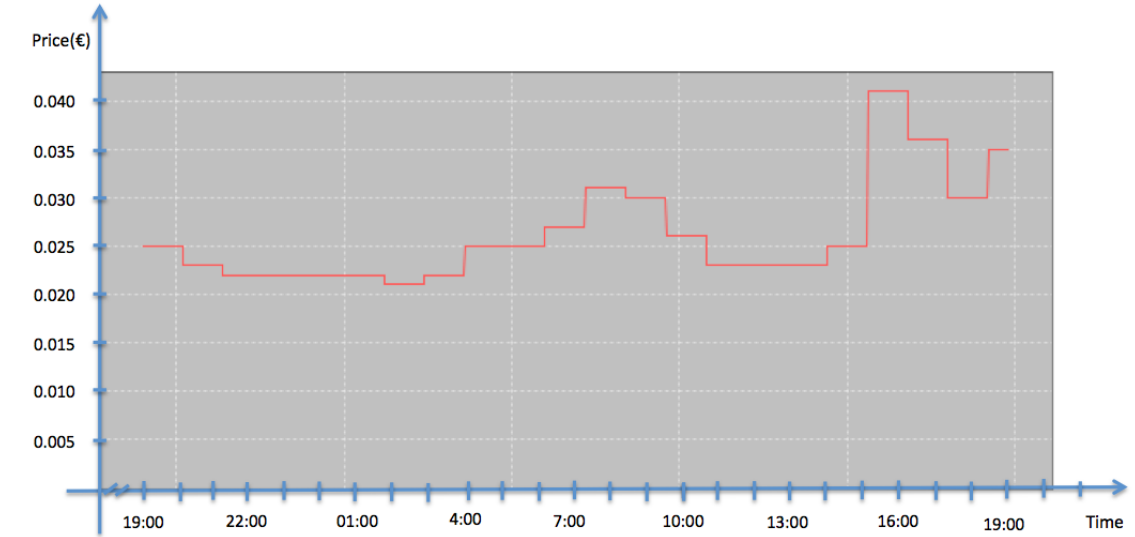


Figure 4.1: Pricing Curve

As depicted in Figure 4.1 the pricing is cheaper during night times when typically industrial and domestic consumption is lower. As seen in the literature review, during the night the prices are cheaper enabling the scheduler to shift tasks and save money and in a grid global sense augmenting the base demand, which is the goal of producers.

This curve was chosen because it represents a good example by showing some common features of a pricing curve such as price differentiation, that can be very small, (as small as 1 cent) but, with a good schedule it enables consumers to save money. Price changes typically occur in the beginning of each hour and last until the next hour.

4.1.2 Available Electricity

Available power is another very important information that must be taken into account when scheduling. This curve 4.2 represents the remaining electricity that can be used for schedulable tasks. The difference between the contracted power and the available energy represents the energy that normally is used for non-schedulable tasks.

In order to make our tests more realistic, contracted power has a standard value for domestic use, which is in this scenario 6kVA. To test more intensively the scheduling algorithm, the available power curve is more demanding than what should be expected in a real world case. The rationale behind this choice was to test the scheduling algorithm in harsh situations, which will

Testing and Results

give us more significant results.

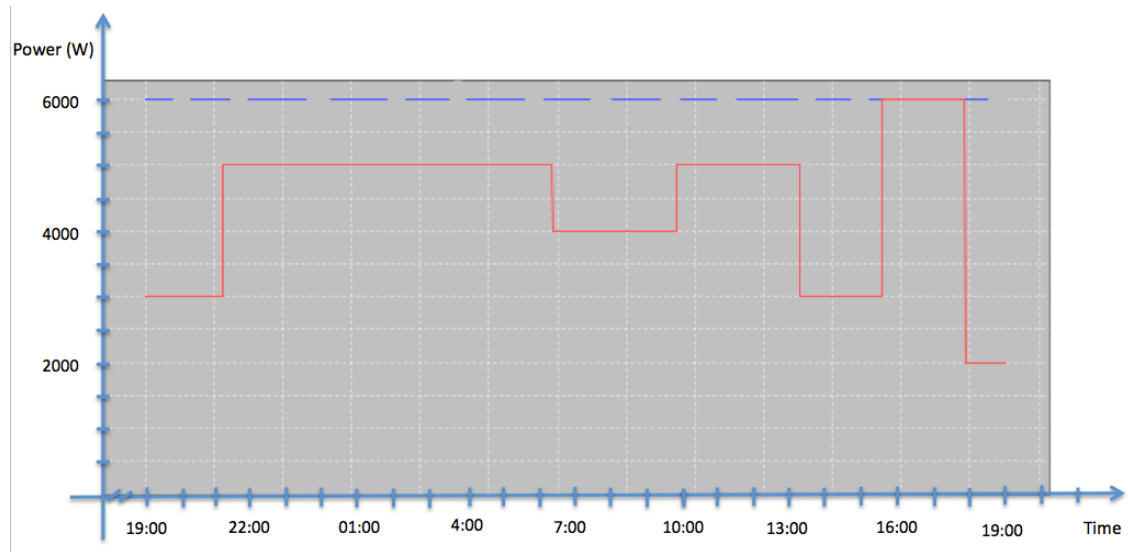


Figure 4.2: Electricity Curve

In Figure 4.2, the slashed line represents contracted power and the continuous line represents the maximum energy that can be used for schedulable tasks.

4.1.3 Devices

In this we explain the characteristics of each device. For the sake of these tests, the program that each device runs is the same, for instance, in every test that includes a “Washing Machine”, will be used the same “machine program” since each program is related to a consumption curve that can be different in case of different programs. For example, with a washing machine, the 90°C wash has a different execution time and consumption curve comparing to the 30°C wash.

Next we explain in depth each device and their characteristics.

Microwaves

This device represents a standard kitchen microwaves. Typically this device uses a considerable amount of energy for a very short time. In each test scenario, the microwaves has the characteristics shown in Table 4.1.

Name	Duration	Preemptive	Consumption Curve
Microwaves	10 minutes	No	Constant at 800 W

Table 4.1: Microwaves

Dish Washer

This device information is represented in Table 4.2. This task can be divided into 3 different tasks, enabling the possibility of scheduling it in different start times to take advantage of fluctuating prices. It also contains a consumption curve, meaning that consumption is not constant.

Name	Duration	Preemptive	Consumption Curve
Dish Washer	90 minutes	Yes. Three 30 min. intervals.	Yes, Figure 4.3

Table 4.2: Dish Washer

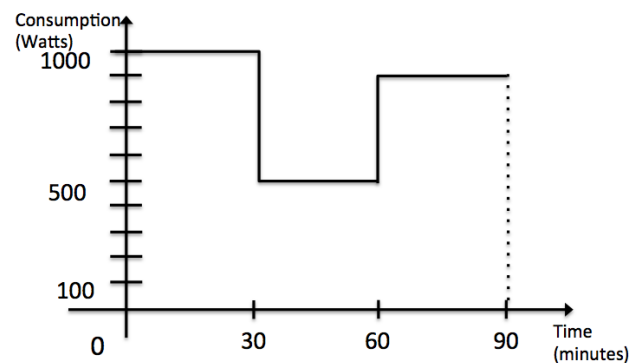


Figure 4.3: Dish Washer Consumption Curve

Washing Machine

This device, which represents a washing machine, is a non-preemptive device that has an associated consumption curve.

Name	Duration	Preemptive	Consumption Curve
Washing Machine	120 minutes	No	Yes, Figure 4.4

Table 4.3: Washing Machine

Oven

This device represents a kitchen oven. Also, it is a non-preemptive device and its consumption is constant during usage.

Name	Duration	Preemptive	Consumption Curve
Oven	45 minutes	No	Constant at 1000 W

Table 4.4: Oven

Testing and Results

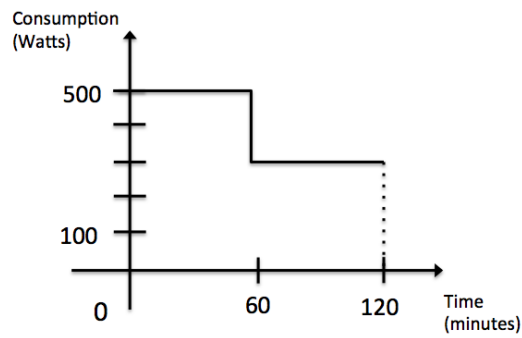


Figure 4.4: Washing Machine Consumption Curve.

Drying Machine

This represents a drying machine, which is a non-preemptive device with a consumption curve associated. The information related to this device are in Table 4.5.

Name	Duration	Preemptive	Consumption Curve
Drying Machine	120 minutes	No	Yes, figure 4.5

Table 4.5: Drying Machine

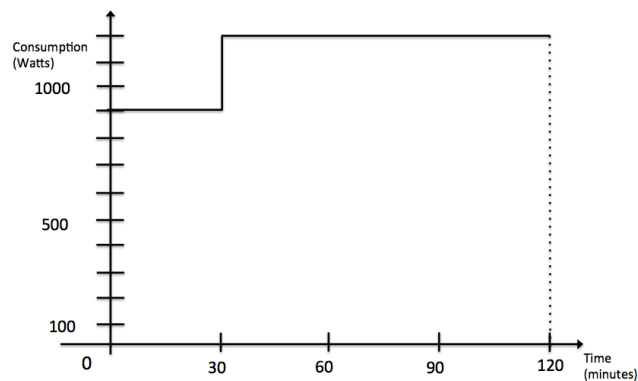


Figure 4.5: Drying Machine Consumption Curve and Task's Preemptive Division.

Electricity Storage Device

This represent a device that possesses the capability to store energy and then use it whenever needed. This represents any device that has an energy storage system, such as batteries. Despite this is a storage device, for the sake of this work, this device does not supply energy to other devices. See Table 4.6.

Name	Duration	Preemptive	Consumption Curve
E.S.D.	300 minutes	Yes. Ten 30 min. intervals.	Constant at 1000 W

Table 4.6: Electricity Storage Device

Air Conditioner

This device represents a air cleaning and conditioning device, which is neither preemptive nor has a variable energy consumption as can be seen in 4.7.

Name	Duration	Preemptive	Consumption Curve
Air Conditioner	10 minutes	No	Constant at 1000 W

Table 4.7: Air Conditioner

Heat Pump

This device represents a water heating device. Its characteristics are in Table 4.8

Name	Duration	Preemptive	Consumption Curve
Heat Pump	60 minutes	No	Constant at 2000 W

Table 4.8: Heat Pump

4.2 Schedule Quality

In this section we present the factors that measure, in this research, the quality of a schedule. In Chapter 3 we have explained the need for producing good schedule solutions with the use of a computer that possesses small computation capabilities. Alongside with this processing restrictions, there is the need for producing schedules as quickly as possible. The fact of this system being an online live system demands quick schedules otherwise normal functioning of devices may be harmed. Knowing this, a compilation of factors must be made in order to evaluate the performance of the scheduling algorithm. The factors are:

- Execution time in milliseconds;
- Proximity to the optimal solution in each case;
- Evolution of solutions during time;
- Standard Deviation of results for the same scenario;

Being defined the factors that measure obtained solutions, next section summarizes the characteristics of the testing machine as well as the tests performed.

4.3 Tests

As said previously, this algorithm will be running on a few resources machine, which implies testing in a computer that has few processing capabilities. The following tests will be running on a 6 year-old laptop that has the following specifications:

- **Operating System** - MacOS X v10.7.5, Lion;
- **Processor** - 2.0GHz, Core2Duo;
- **Memory** - 2Gb DDR3, 1067Mhz;
- **Hard Drive** - 320Gb, SATA 2 at 7200rpm.

These specifications are very similar in processing power as the machine that will run the scheduler when it reaches production phase. The specifications are [CuB13]:

- **Operating System** - Light Linux Distribution. Custom compilation for this project.;
- **Processor** - 0.8GHz, ARM;
- **Memory** - 2Gb DDR3.
- **Hard Drive** - Solid State Drive.

In this section will be given focus on tests that were performed in order to understand how the developed algorithm behaves in terms of quality of solution and speed.

This section contains a relevant set of tests that prove the performance and adaptability of proposed algorithm to meet the goals proposed in this dissertation. A summary of each schedule quality is also made by observing the factors explained in 4.2. Each test case has a user input associated and this input represents the willingness of consumer to extend, postpone or simply program a device execution to run in a certain time frame.

An scheduling algorithm execution can be seen as a re-schedule if pricing or available electricity change value. This is true because, on every change of price, task schedule or available power new schedule must be created in order to adapt to recent changes. This means that, new input from consumer or any order from the grid is contemplated in this batch of tests.

Each test in the sequel represents an increase of difficulty on generating good schedules, not only by increasing the number of tasks but also by changing the number of preemptive tasks added to the schedule and the number of tasks that have a variable consumption curve. These two factors increase the number of possible solutions by large numbers.

4.3.1 Test 1

In this test case is simulated the introduction of 7 new non-preemptive tasks. These tasks also have a constant consumption of energy. These are the specifications of each task added, each one having baseline and deadline introduced by the user. The duration value is an user input in all the tests since there was no possibility to test this algorithm in a real *Smart Grid*.

Table 4.9: Test 1 User's Input

Name	Baseline	Deadline	Duration
Microwaves	19:00	22:30	10 min
Microwaves	16:15	19:00	10 min
Oven	19:06	21:45	45 min
Oven	10:00	13:00	45 min
Heat Pump	3:00	9:00	60 min
Heat Pump	3:00	9:00	60 min
Air Conditioner	8:30	12:50	10 min

Table 4.9 shows user's input on deadline and baseline for each task. This schedule has the optimal cost of 0.1329€. If the tasks were scheduled to their baselines, apart from the inconvenience cost of turning each device on at given hours, the user would have to manage the power conflict between the two heat pump executions, that cannot be running at the same time otherwise there would be an energetic shutdown because the needed energy to run these devices is above the available energy. With this being said, pricing for this set of tasks without scheduling would be 0.14985€.

This presented scenario is very common nowadays in domestic environments without a smart grid, by addressing often used devices in their normal time of use (ex: Oven at mealtimes or water pump for morning shower). It is also a simple scenario in terms of computation. Despite that, this scenario can generate a search space of 176802912000000 schedules just by combining the tasks, if we disregard the available power curve.

This previous value is calculated by multiplying the number of possible start times of all tasks, obtaining the search space and not the number of possible solutions; this can decrease if available power has a restrictive power curve.

4.3.1.1 Computed Solution

For each performed test is calculated a Gantt chart that contains the representation of each task, as well as its start time and end time. Figure 4.6 shows the optimal Gantt chart for this first test, which is also our algorithm's output.

Another important information regarding this test is consumed energy versus the available energy for running this test case scenario. Figure 4.12 shows the used power versus available power.

Testing and Results

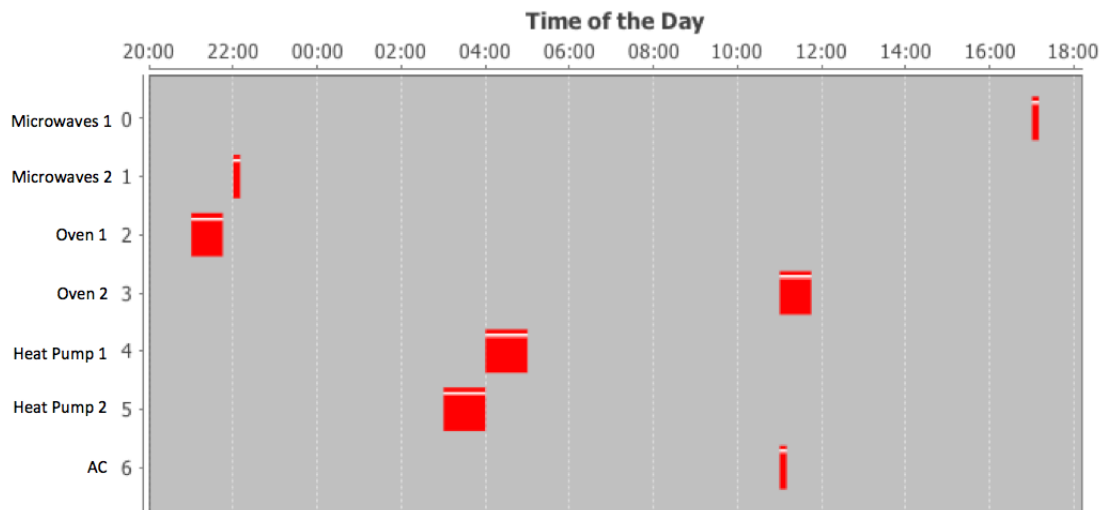


Figure 4.6: Algorithm's Output

Note that during two hour consumption is equal to the available power, making the optimal solution the cheapest regarding available power. On this schedule there is a conflict between two executions of the same devices that user has chosen to give the same baseline and deadline. This fact obliged the scheduling algorithm to make these tasks run one after another in order to not exceed the available power, avoiding electricity shortage and consequent shutdown of all devices.

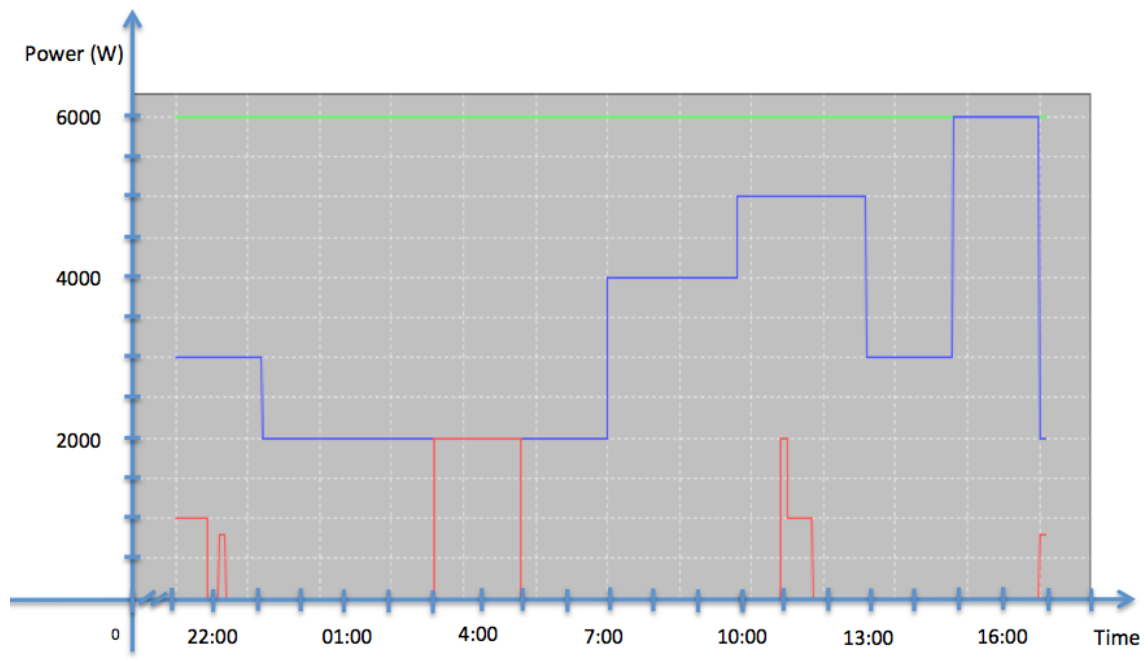


Figure 4.7: Available Power and Consumed Electricity by Test 1 Schedule

Testing and Results

For this test was used a time limit of one second to find the best solution possible for the given problem. The scheduling algorithm ran for 100 times and outputted the solution at each execution.

4.3.1.2 Analysis

The execution of 100 times this test case produced the following information. Time limit for this test's execution was 2 seconds.

Number of Optimal Solutions Found	100 / 100 = 100%	
Standard Deviation	0 €	
Average Execution Time	55.77 milliseconds	
Minimum Optimal Find Time	0 milliseconds	
Maximum Optimal Find Time	197 milliseconds	
Optimal Solutions Found	Until 10 ms	12
	Until 100 ms	85 (+ 73)
	Until 1000 ms	100 (+ 15)

Table 4.10: Numeric Results of 100 Executions of Test 1.

Table 4.10 shows that for all 100 executions the scheduler returned 0.1329€, being the optimal solution for this scenario. This fact means that the scheduler algorithm, for this scenario has an accuracy of 100%.

While saving the results, it was also observed execution time on each run of the scheduler. The average time on finding solutions for all these 100 executions was 55.77 milliseconds, being the highest execution time of 197 milliseconds and the lowest of 0 milliseconds, meaning that the optimal solution was found while generating the initial population.

Table 4.10 also shows that if algorithm execution time was reduced to 100 milliseconds it would have also a very good ratio of optimal results: 85% of times the scheduler would return an optimal solution.

4.3.2 Test 2

In this test case is simulated the introduction of 7 new tasks by the user. This new schedule contains 2 preemptive tasks with consumption curve and 2 more tasks that only have a consumption curve (non-preemptive tasks). Remaining tasks presented on the schedule are non-preemptive with constant consumption tasks. These are the specifications of each task added, each one having baseline and deadline introduced by the user. Running time in order to reach optimal solution in this test is 5 seconds.

For the input described in Table 4.11, optimal solution is 0.146€. For this specific test, if the tasks were scheduled to their baseline the cost of this schedule would be 0.1545€.

This scenario, despite having the same number of tasks, is far more complex than Test 1. This fact is induced by preemptive tasks and consumption curve associated with some of added devices. As said previously 3.1.1 each preemptive task is a set of non preemptive tasks, therefore making

Testing and Results

Name	Baseline	Deadline	Duration
Microwaves	16:15	19:00	10 min
Dish Washer	22:45	8:10	45 min
Dish Washer	14:00	19:00	45 min
Oven	19:06	21:45	45 min
Heat Pump	3:00	9:00	60 min
Washing Machine	19:00	11:00	120 min
Air Conditioner	17:00	18:50	10 min

Table 4.11: Test 2 User's Input

this schedule a 11 task schedule because each “Dish Washer” execution adds 3 tasks instead of only 1. Also the correlation between them adds another complexity by having precedence failures when second and third tasks start before the end of their previous tasks. Another factor that contributes to complexity is consumption curves. Consumption curves make even more complex finding best combinations in order to fully take advantage of cheaper time frames, for each task.

4.3.2.1 Computed Solution

The Gantt chart depicted in Figure 4.8 shows the optimal schedule for this test. In this optimal solution can be observed that one preemptive task (first execution of the “Washing Machine”) is executed continuously, while the second preemptive task execution of the same device stopped between second and third sub-task to avoid a more expensive electricity price.

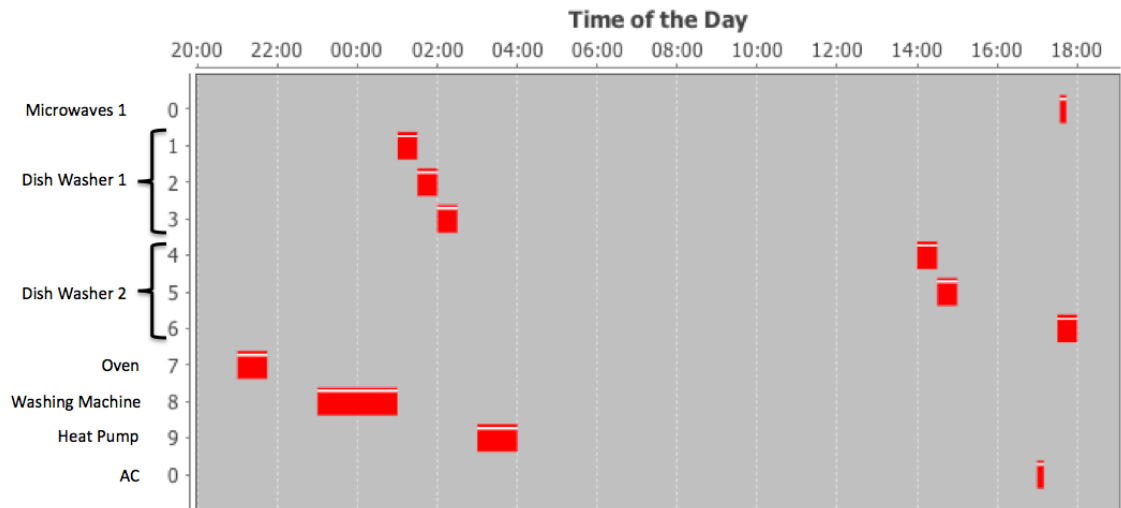


Figure 4.8: Results Within 4 Seconds Time Limit.

This test case enables the appearance of a typical difficulty in finding optimal solution when devices have a consumption curve. In Figure 4.9 is compared the optimal consumption curve

versus a near optimal consumption curve that is worse than optimal for 1 cent. This more expensive curve was obtained by stopping the scheduling algorithm before finding optimal solution.

As depicted in the next two consumption curves 4.9, the difference between the two solutions, optimal and second best resides on the difference of availed energy. The optimal chooses to address the 2000 Watt device to cheapest hour and the second one chooses to divide the usage between the “Washing machine” and “Oven”.

Despite the fact that the highest consumption time of the “Washing machine” was addressed to the cheapest price, it would be more advantageous to anticipate this execution in order to clear time frame to the “Oven” execution which would take more advantage by using all available energy. This conflict is caused by the small electricity price difference between the two time frames.

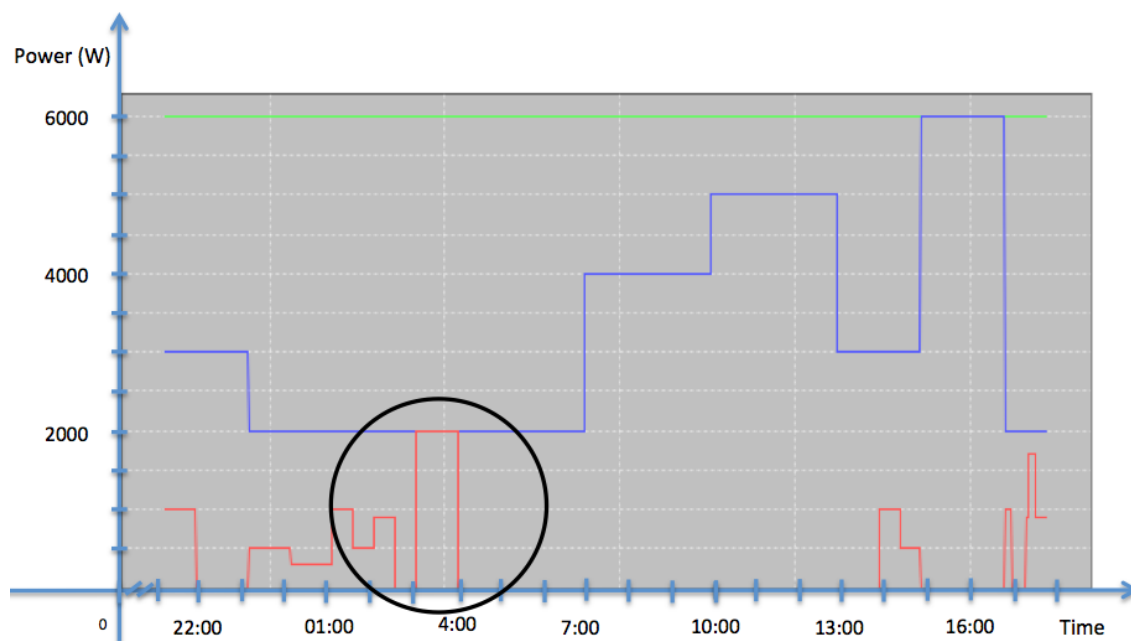


Figure 4.9: Optimal Consumption Curve.

What happens is that a device, particularly the “Washing Machine” execution that has a crescent consumption curve and does not take full advantage of the 2000 Watts available at cheapest price. Also, because there is no available power for all tasks to run between 3 am and 4 am, there is concurrency. The optimal solution has precisely this difference, it shifts less consuming task to a higher price and puts the 2000 Watt Heat Pump 1 hour duration task to take full advantage of cheapest price.

4.3.2.2 Analysis

The execution of 100 times this test case produced the following information.

Table 4.12 shows that for this scenario, the accuracy of finding optimal solution is 89% having a standard deviation of nearly 0.0001€. This test also has shown that the majority of times, the

Testing and Results

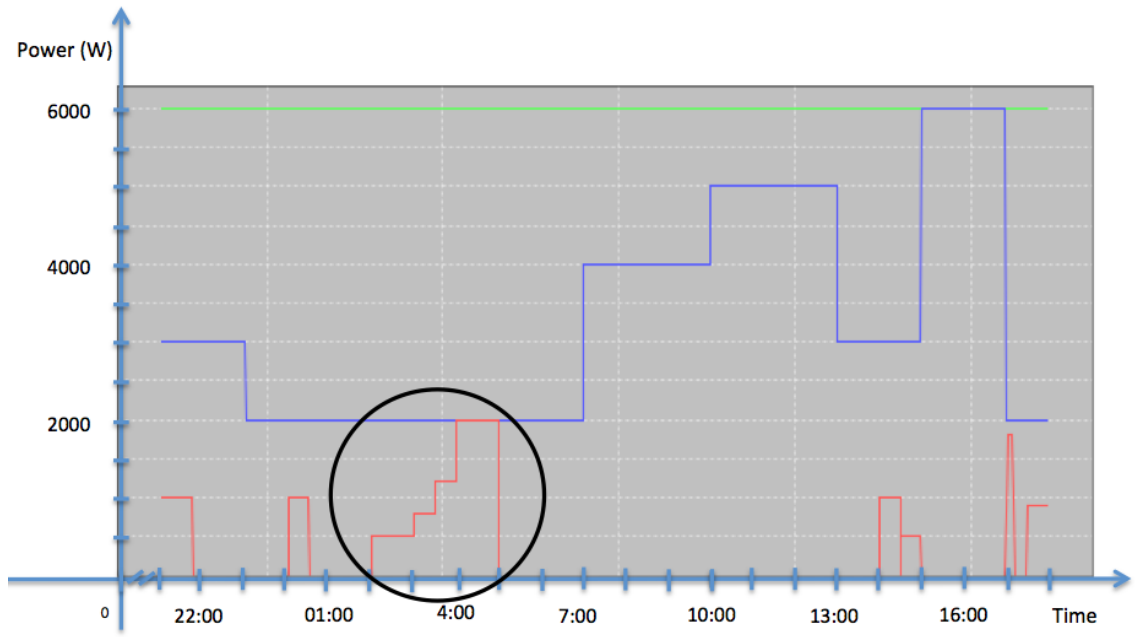


Figure 4.10: Near Optimal Consumption Curve.

Number of Optimal Solutions Found	89 / 100 = 89%	
Standard Deviation	0.000123	
Average Execution Time	680.63 milliseconds	
Minimum Optimal Find Time	3968 milliseconds	
Maximum Optimal Find Time	10 milliseconds	
Optimal Solutions Found	Until 1000 ms	74
	Until 3000 ms	84 (+ 10)
	Until 5000 ms	89 (+ 5)

Table 4.12: Numeric Results of 100 Executions of Test 2.

optimal solution is reached before 1 second of execution. In contrast to the previous test, none of initial populations contained an optimal solution.

4.3.3 Test 3

In this test is simulated the introduction of 16 tasks, that simulate a very busy household, with a continuous climate management. This scenario can somehow simulate a weekend setup, when typically people use their more appliances even though in a more diluted way. This test will run during 10 seconds in order to reach the optimal solution on each execution.

For the input described in Table 4.13, optimal solution is 0.24649 €. If the start time for each task was set as its baseline and if it was ignored the fact that 64 minutes of consumption is above available energy between an “Air Conditioner” execution and “Drying Machine” for 4 minutes and two “Heat Pump” executions for 60 minutes, the final cost would be 0.04675€ more expensive, placing the price at 0.29324€.

Testing and Results

Table 4.13: Test 3 User's Input

Name	Baseline	Deadline	Duration
Microwaves	16:15	19:00	10 min
Microwaves	19:00	22:30	10 min
Dish Washer	22:45	8:10	45 min
Dish Washer	14:00	19:00	45 min
Drying Machine	19:00	7:30	45 min
Oven	19:06	21:45	45 min
Oven	10:00	13:00	45 min
Heat Pump	3:00	9:00	60 min
Heat Pump	3:00	9:00	60 min
Washing Machine	19:00	11:00	120 min
Air Conditioner	17:00	18:50	10 min
Air Conditioner	14:00	15:00	10 min
Air Conditioner	13:00	15:00	10 min
Air Conditioner	8:30	12:50	10 min
Air Conditioner	22:00	00:00	10 min
Air Conditioner	20:00	22:00	10 min

This third test proposes a rather complex and filled schedule that tries to emulate a very busy house with a high usage of appliances. In consequence of this highly busy schedule the search space is gigantic, making the search for optimal solution a very difficult task. Also, some of the tasks are preemptive and some have a variable consumption curve. Despite being presented 16 tasks, they represent 22 tasks in total by unfolding preemptive tasks.

4.3.3.1 Computed Solution

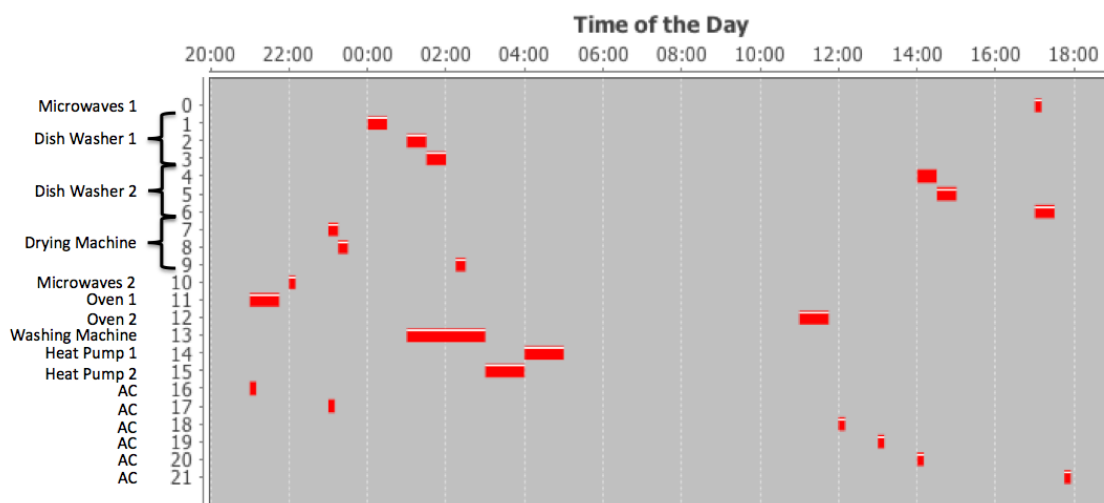


Figure 4.11: Optimal Scheduling Found.

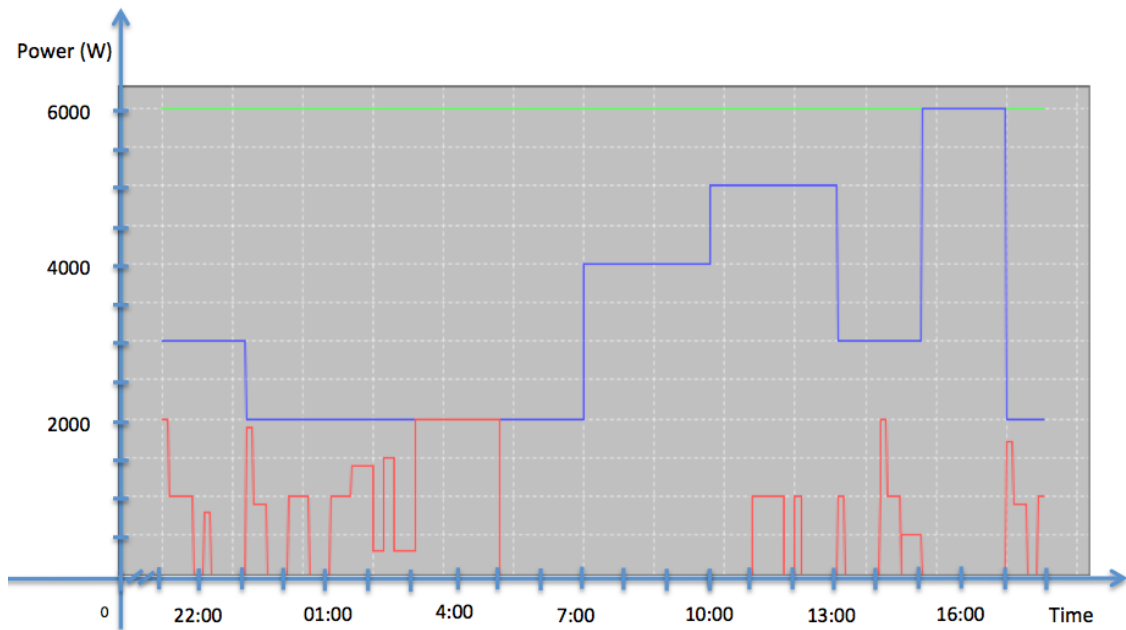


Figure 4.12: Optimal Consumption Curve.

4.3.3.2 Analysis

The execution of 100 times this test case produced the following information.

Number of Optimal Solutions Found	86 / 100 = 86%	
Standard Deviation	0.0008154	
Average Execution Time	3796.95 milliseconds	
Minimum Optimal Find Time	13523 milliseconds	
Maximum Optimal Find Time	409 milliseconds	
Optimal Solutions Found	Until 1000 ms	11
	Until 3000 ms	30 (+ 19)
	Until 5000 ms	66 (+ 36)
	After 5000 ms	86 (+ 20)

Table 4.14: Numeric Results of 100 Executions of Test 3.

In this test case scenario the number of optimal solutions is 86% and the standard deviation is a little above 0.0008€. As can be noticed from previous tests, this one is clearly slower in finding the optimal solution despite the minimum time be near 400 milliseconds. Near 25% of optimal solutions only were found after the 5 second mark which makes possible 66% of times a optimal solution under 5 seconds, which is this research maximum execution time.

4.3.4 Test 4

This fourth test simulates the introduction of 5 tasks that were previously tested, which some of them have consumption curves and preemption but also have an introduction of a peculiar task,

Testing and Results

the “Electricity Storage Device”. As said previously this device is compared almost to a battery, which stores energy for its further consumption. This device runs for a very long time and has a 24 time window and is also preemptive, dividing itself in 10 different tasks. This test execution will have a maximum time of 15 seconds for find optimal solution.

Table 4.15: Test 4 User’s Input

Name	Baseline	Deadline	Duration
Microwaves	19:00	20:00	10 min
Microwaves	18:00	19:00	10 min
Drying Machine	19:00	23:00	45 min
Washing Machine	13:00	16:00	120 min
Oven	9:00	12:00	45 min
E.S.D.	19:00	19:00	300 min

Table 4.15 shows user’s input for each task. This schedule has the optimal cost of 1.72379€ . The cost would be 2.23775€ if for each task the start time was its baseline. In this test we test how the algorithm behaves in an extreme preemptive case.

Despite having a small number of devices to schedule, preemptive devices turn this example into a complex scenario by creating 17 tasks to be scheduled by the algorithm. The biggest challenge in this scenario is related to how does the schedule handle so many interdependent tasks. The best solution found can be seen below.

4.3.4.1 Computed Solution

Figure 4.13 chart shows the best solution found, which is not the optimal solution for this problem. This solution, much like the other solutions found in this test case struggled with the correlation of “E.S.D.”’s tasks which didn’t allow the schedule to find the best solution for this scenario. Despite that fact, all 100 solutions were very near optimal. The optimal solution would be reached if this 5 hour task (“Electricity Storage Device”) started at 0 am and ended at 5 am, taking advantage of the cheapest time frames in the 24 hour pricing curve.

This consumption curve, shows that there is still much energy that can be used at every minute during the 24 hours. So, there was not much power restrictions in order to schedule this task into optimal time frames.

4.3.4.2 Analysis

As said previously, this test did not find any optimal solution. Despite that, in all 100 tests the other tasks were 100% accurate in finding optimal solution. Standard deviation for this example shows that most of results were near optimal which is 0.172379€ . Average execution time for each solution is decent, but not as good as the other 3 tests, which is explained by the processing time used by this “special” task, the 10 sub-task “Electricity Storage Device”.

Testing and Results

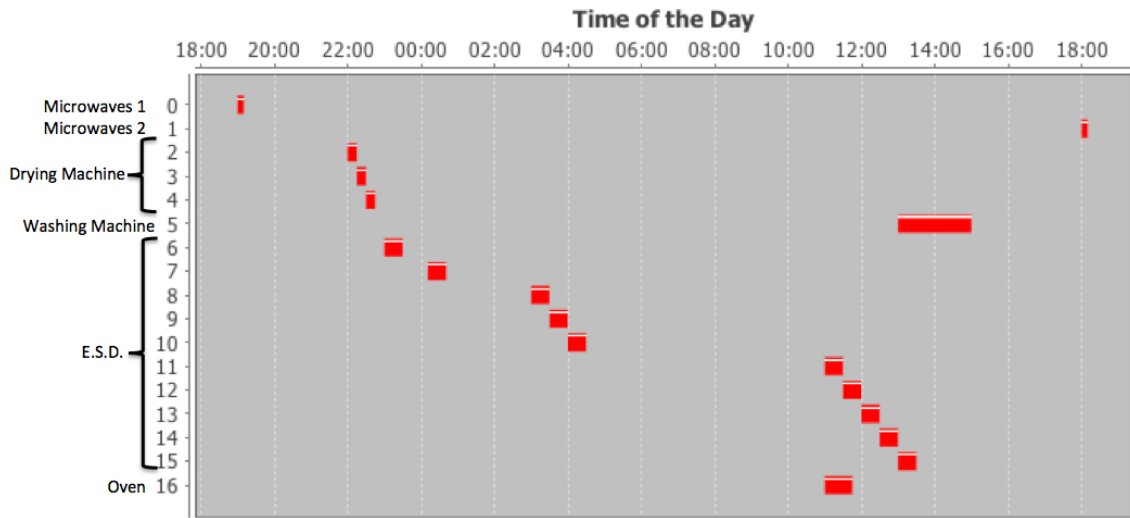


Figure 4.13: Results Within 4 Seconds Time Limit.

Number of Optimal Solutions Found	0 / 100 = 0%	
Standard Deviation	0.025363	
Average Execution Time	2348.91 milliseconds	
Minimum Optimal Find Time	∞ milliseconds	
Maximum Optimal Find Time	∞ milliseconds	
Optimal Solutions Found	Until 1000 ms	0
	Until 3000 ms	0 (+ 0)
	Until 5000 ms	0 (+ 0)
	After 5000 ms	0 (+ 0)

Table 4.16: Numeric Results of 100 Executions of Test 4.

4.3.5 Test 5

In this last test is simulated many random schedule introductions by the user. For this, a testing framework was created in order to achieve quantifiable results.

- **Number of Tasks** - Between 6 and 10 tasks;
- **Preemptive Devices** - Between 50% of devices in each schedule must be preemptive;
- **Devices With Variable Consumption Curve** - 50% devices in each schedule must have a variable consumption curve;
- **Maximum Execution Time of the Scheduler** - 3 seconds.

Previous scenario characteristics show in our opinion a good example of how a user configures a typical household schedule. This test will perform 50 different schedules, generated randomly, taking into account the characteristics defined above. Also each task refers to any of previous

explained devices. Each baseline and deadline are generated randomly, as long as the task can run within the available time defined by these two parameters.

This test brings a complexity level previously tested. Nevertheless in this test is also tested not only the number of tasks per schedule and their characteristics, but the coherence of the researched algorithm in delivering good solutions in random inputs by the users.

4.3.5.1 Analysis

Number of Optimal Solutions Found	2 / 50 = 96%
Overall Standard Deviation	0.00016€
Overall Average Execution Time	142.103 ms
10 Tasks	
Optimal Solutions Found	7/8 = 87.5%
Standard Deviation	0.0006€
Average Execution Time	267.125 ms
9 Tasks	
Optimal Solutions Found	10/11 = 90.9%
Standard Deviation	0.0002€
Average Execution Time	260.5 ms
8 Tasks	
Optimal Solutions Found	11/11 = 100%
Standard Deviation	0€
Average Execution Time	258.09 ms
7 Tasks	
Optimal Solutions Found	10/10 = 100%
Standard Deviation	0€
Average Execution Time	20.8 ms
6 Tasks	
Optimal Solutions Found	10/10 = 100%
Standard Deviation	0€
Average Execution Time	4.0 ms

Table 4.17: Numeric Results of 50 Executions of Test 5.

These results show high percentage of optimal results in all possible cases within the proposed scenario. It also can be observed that average execution time increases as the number of tasks to schedule rise. In all tests that have less than 8 tasks to schedule all results obtained are optimal and above that number of tasks, optimal results are above 87%.

4.4 Results Analysis

Previous tests have shown different schedules that are representative of noticeable cases in this problem. Overall the results obtained were very satisfying and reached the goals proposed. This

Testing and Results

schedule can deliver relatively fast solutions with a high rate of them being optimal. Test 1 has shown that for its scenario, which is believed that these scenarios are the most directly related to everyday use, this algorithm can deliver 100% of time the optimal results for the input schedule. This fact is seen as an excellent result, making this approach reliable and proving this implementation can handle easily these type of problems within this relatively small search space.

The second test introduced complex consumption curves and preemptive tasks. This algorithm has proven itself very accurate in this type of situations. Although it did not reach the optimal solution in every single execution, it reached a consistent near optimal when not reaching optimal as can be seen by this test's standard deviation. This test case can be seen as a common test case because it does not test preemptive tasks with many chunks, since it may not be possible to divide any task in more than 2 or 3 tasks due to devices' specifications and execution matters.

On third test case scenario, despite the fact that 86% of solutions were optimal, there is a noticeable amount of these solutions that were found after 5000 milliseconds, which is somehow a concern about the strength of the algorithm. To justify this fact, it must be said that this is related to the gigantic search space associated and the number of tasks that are, in a real life situation, a very extreme scenario. Despite that, other non-optimal solutions were very close to optimal proving again the consistency of the developed algorithm.

Test case 4 shows one flaw of this approach. This flaw is related to preemptive tasks and the number of chunks of these tasks. As can be observed, in this test is introduced a 10 part task that in none of 100 tests performed, made possible to reach a solution. This comes from the fact that each of these 10 tasks depend on each other to choose their optimal start time, and the quantity of them makes almost impossible to reach the optimal solution since they struggle between them to reach an optimal solution. This does not happen with less divided preemptive tasks because they produce a smaller search space that enables the genetic algorithm to understand what can be the best solution for each part in order to generate the best overall schedule. What happens in this test case is that the first chunks find the best individual solution, leaving others to third or fourth best solutions which increases final price.

The final test has proven that even when simulating random inputs from the user, the algorithm can deliver very good results within 3 seconds, which was the time limit for computing the best solution possible. With this test is now possible to assert with more confidence that the proposed algorithm is ready for testing in a real *Smart Grid* scenario.

Testing and Results

Chapter 5

Conclusions

Based on related work, this research took those approaches' best argument and took a step forward on reducing peak demand with aid of the *Smart Grid's* infrastructure. To achieve that, this research pursued the objective of creating a scheduling system that enabled the control of when each home device uses energy. This control is made by taking into account user's preferences. These preferences relate to when consumers need the device executions to be over.

The scheduler implemented uses an *Evolutionary Algorithm* approach. This approach enables the system to run quickly without compromising optimality in the solution. In every heuristic implementation, was present the goal of designing them as light as possible in terms of processing needs since the device that will run this algorithm has very low processing power.

Meeting the goals with the final work it can be concluded that this work studied the best approaches in energy scheduling problems. Identifying the capabilities of a *Smart Grid* has open new perspectives to approach this research such as the existence of *Smart Meter* that became targeted to be the processing unit. Literature review has shown very different approaches to different algorithms which were adapted in order to schedule power usage, but despite that, there was found that each approach lacked terms of realism by the way of how each problem was addressed. Soon it was observed that main improvements would come from the approach to the problem and how to combine all factors in one system, and not too much related to the type of algorithm that could be used.

Main Contributions

Despite the need of testing extensively the developed algorithm, it is straight to conclude that our main hypothesis is confirmed positively. This research proved, in this conceptual experience, that it is possible to control and schedule power consumption in households. It is possible to manage the energy consumption by monitoring each device to take advantage of electricity pricing without violating hard constraints such as available power and task integrity. to achieve that there must be available device and *Smart Grid* information such as consumption of each device and its characteristics, as well as information about the prices that is provided by producers. With this

Conclusions

being said, it is possible to confirm that is achievable the reduction of peak demand using all factors addressed previously.

Tests have shown that this algorithm can produce very good results with all kinds of household scenarios with a high rate of optimal solutions or very near them, as standard deviation shows. Another noticeable characteristic of this solution is execution time. For a considerable amount of tasks it can provide an optimal solution almost instantaneously, which proves to be a very good achievement in live systems. But there is also room for improvement, as can be read on Section 5.1.

5.1 Future Work

The completion of goals in this research has shown huge potential in future work. Since this researched algorithm will be present in EnAware, it must be tested to exhaustion with real devices and in a real smart grid environment. With these tests it will be guaranteed the performance shown in tests section.

This algorithm can easily be expanded to use not only electricity provided by producers. Electrical cars, solar panels or external batteries provide very good alternatives to *Smart Grid* as sources of energy. Also, related to electrical cars, they can be used as special batteries to provide energy when available energy is not enough, helping consumers to carry on their consumption without having electricity shutdown or, in an extreme case, have to renegotiate contracted power with their provider. Also there will be the need to manage how produced energy can be spent since when there is no more storage capacity to keep the energy, sell it to the grid taking advantage of its bidirectional power transmission.

Also, related to data mining of device consumption readings, it would be necessary to develop an more accurate algorithm that can predict with more certainty how much energy will be available for schedulable tasks. This information can be also used to create a contracted power suggesting system that could inform consumers to increase or reduce their contracted power in order to prevent energy shortage or to save money.

Related to devices it would be very interesting create a “Driver” system that permits the easy introduction of new devices in the system, by containing for each program its consumption curve and preemptive intervals, if any.

References

- [aI13] Tyler at InterconGreen. Available at: <http://intercongreen.com/>, June 2013.
- [All13a] United Nations Sustainable Energy For All. Available at: <http://www.sustainableenergyforall.org/about-us>, June 2013.
- [All13b] OSGi Alliance. Available at: <http://www.osgi.org/Technology/WhyOSGi>, January 2013.
- [BZ13] Marilyn A. Brown and Shan Zhou. Smart-grid policies: an international review. *Wiley Interdisciplinary Reviews: Energy and Environment*, 2(2):121–139, 2013.
- [Cen13] Newcastle University Engineering Design Center. Available at: <http://www.edc.ncl.ac.uk/highlight/rhjanuary2007g02.php>, June 2013.
- [CKV11] Chen Chen, Shaline Kishore, and Lawrence V.Snyder. An innovative rtp-based residential power scheduling scheme for smart grids. 2011.
- [CKW⁺12] S.C. Chan, K.M.Tsui, H.C. Wu, Yunhe Hou, Yik-Chung Wu, and Felix F. Wu. Load-/price forecasting and managing demand response for smart grids. *IEEE signal processing magazine*, pages 68–85, September 2012.
- [Com08] Litos Strategic Communication. The smart grid: An introduction. Technical report, U.S. Department of Energy, October 2008.
- [CuB13] SolidRun CuBox. Available at: <http://www.solid-run.com/cubox>, June 2013.
- [DD08] G. Deconinck and B. Decroix. Smart metering tariff schemes combined with distributed energy resources. 2008.
- [dP13] Electricidade de Portugal. Available at: <http://www.edp.pt/pt/negocios/apoioaocliente/Pages/TarifaBiHoraria.aspx>, January 2013.
- [EMCA⁺11] E.Matallanas, M.Castillo-Cagigal, A.Gutiérrez, F.Monsanterio-Huelin, E.Caamaño-Martín, D.Masa, and J.Jiménez-Leube. Neural network controller for active demand-side management with pv energy in the residential sector. 2011.
- [Eur13] European Commission Intelligent Commission Europe. Available at: http://ec.europa.eu/energy/intelligent/getting-funds/financing-streams/index_en.htm, June 2013.
- [HG10] Katherine Hamilton and Neel Gulthar. Taking demand response to the next level. *IEEE power & energy magazine*, pages 61–62, May / June 2010.

REFERENCES

- [JS92] KennethA. Jong and WilliamM. Spears. A formal analysis of the role of multi-point crossover in genetic algorithms. *Annals of Mathematics and Artificial Intelligence*, 5(1):1–26, 1992.
- [Kri10] Rajamani Krishnan. Meters of tomorrow. *IEEE power & energy magazine*, pages 96–94, May/June 2010.
- [KT11] Iordanis Koutsopoulos and Leandros Tassitulas. Control and optimization meet the smart power grid:scheduling of power demands for optimal energy management. 2011.
- [LL12] Adam Lipowski and Dorota Lipowska. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, 391(6):2193 – 2196, 2012.
- [LSS12] Thillainathan Logenthiran, Dipti Srinivasan, and Tan Zong Shun. Demand side management in smart grid using heuristic optimization. 2012.
- [MM09] P. McDaniel and S. McLaughlin. Security and privacy challenges in the smart grid. *Security Privacy, IEEE*, 7(3):75–77, 2009.
- [RW08] Peter C. Reiss and Matthew W. White. What changes energy consumption? prices and public pressures. *The RAND Journal of Economics*, 39(3):636–663, 2008.
- [STYP10] Von-Wun Soo, Men-Shen Tsai, Wan-Yu Yu, and Yen-Bo Peng. Coordination of a society of agents for automatic distribution system restoration:toward a smart grid. 2010.
- [Sys13] MIT Portugal Sustainable Energy Systems. Available at: <http://www.mitportugal.org/sustainable-energy-systems/>, June 2013.
- [TdMM06] James W. Taylor, Lilian M. de Menezes, and Patrick E. McSharry. A comparison of univariate methods for forecasting electricity demand up to a day ahead. *International Journal of Forecasting*, 22(1):1 – 16, 2006.
- [Wee13] European Comission-Sustainable Energy Week. Available at: <http://www.eusew.eu/>, June 2013.
- [ZW11] Alan T. Zehnder and Zellman Warhaft. University collaboration on wind energy. 2011.